(4)

**BBN Systems and Technologies Corporation**

A Subsidiary of Bolt Beranek and Newman Inc.

**AD-A212 368**

BBN Report No. 7138

Final Report
Contract No. N00039-85-C-0423
30 April 1985 - 31 July 1989

# COMBINING MULTIPLE KNOWLEDGE SOURCES FOR CONTINUOUS SPEECH RECOGNITION

R. Schwartz, C. Barry, Y-L. Chow, A. Derr,
O. Kimball, F. Kubala, J. Makhoul

August 1989



Submitted to:

DARPA/ISTO
1400 Wilston Blvd.
Arlington, VA 22209

SPAWAR
Code 613
Washington, D.C. 20306

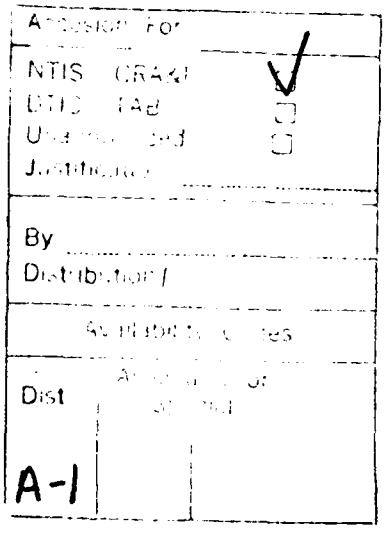
BBN Report No. 7138

Final Report
Contract No. N00039-85-C-0423
30 April 1985 - 31 July 1989

# COMBINING MULTIPLE KNOWLEDGE SOURCES FOR CONTINUOUS SPEECH RECOGNITION

R. Schwartz, C. Barry, Y-L. Chow, A. Derr,
O. Kimball, F. Kubala, John Makhoul

August 1989




Submitted to:

DARPA/ISTO
1400 Wilston Blvd.
Arlington, VA 22209

SPAWAR
Code 613
Washington, D.C. 20306



BBN Report No. 7138

Final Report
Contract No. N00039-85-C-0423
30 April 1985 - 31 July 1989

# COMBINING MULTIPLE KNOWLEDGE SOURCES FOR CONTINUOUS SPEECH RECOGNITION

R. Schwartz, C. Barry, Y-L. Chow, A. Derr,
O. Kimball, F. Kubala, John Makhoul

August 1989

Submitted to:

DARPA/ISTO
1400 Wilston Blvd.
Arlington, VA 22209

SPAWAR
Code 613
Washington, D.C. 20306

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

| REPORT DOCUMENTATION PAGE | | READ INSTRUCTIONS BEFORE COMPLETING FORM |
|---|---|---|
| 1. REPORT NUMBER<br>BBN Report No. 7138 | 2. GOVT ACCESSION NO. | 3. RECIPIENT'S CATALOG NUMBER |
| 4. TITLE (and Subtitle)<br>Combining Multiple Knowledge Sources For Continuous Speech Recognition | | 5. TYPE OF REPORT & PERIOD COVERED<br>Final Report<br>4/30/85 - 7/31/89 |
| | | 6. PERFORMING ORG. REPORT NUMBER<br>7138 |
| 7. AUTHOR(s)<br>R. Schwartz, C. Barry, Y-L. Chow, A.Derr, O. Kimball, F. Kubala, J. Makhoul | | 8. CONTRACT OR GRANT NUMBER(s)<br>N00039-85-C-0423 |
| 9. PERFORMING ORGANIZATION NAME AND ADDRESS<br>BBN Systems and Technologies Corporation<br>10 Moulton Street<br>Cambridge, MA 02138 | | 10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS |
| 11. CONTROLLING OFFICE NAME AND ADDRESS<br>SPAWAR<br>Code 613<br>Washington, D.C. 20306 | | 12. REPORT DATE<br>August 1989<br>13. NUMBER OF PAGES<br>53 |
| 14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) | | 15. SECURITY CLASS. (of this report)<br>Unclassified<br>15a. DECLASSIFICATION/DOWNGRADING SCHEDULE |

16. DISTRIBUTION STATEMENT (of this Report)

Distribution of the document is unlimited. It may be released to the Clearinghouse, Dept. of commerce, for sale to the general public.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

continuous speech recognition, hidden Markov models, language modelling, speech databases, speaker adatpation.

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The objective of this project has been to develop methods and techniques to coordinate the many sources of knowledge in the decision for a continuous speech recognition system. This effort includes finding methods for effectively combining information from various knowledge sources, and for developing recognition search strategies that find the most likely word sequence, given the input speech. These search strategies must consider a very large number of word-sequence hypotheses in a computationally

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

efficient manner. To develop and demonstrate these techniques, we designed and implemented a complete word recognition system for continuous speech which is capable of incorporating knowledge from several sources, including lexical, phonetic, phonological, and grammatical knowledge. The complete system, called BYBLOS, has been shown to achieve the highest recognition accuracy to date on standard government tests using a 1000-word continuous speech corpus.

# Contents

# 1 Executive Summary

This is the final report of a project entitled "Multiple Knowledge Sources for Speech Recognition", sponsored by the Defense Advanced Research Projects Agency (DARPA) and monitored by SPAWAR under Contract N00039-85-C-0423. The project spanned the period 30 April 1985 to 31 July 1989.

The objective of this project has been to develop methods and techniques to coordinate the many sources of knowledge in the decision process for a speech recognition system. This effort includes finding methods for effectively combining information from various knowledge sources, and for developing recognition search strategies that find the most likely word sequence, given the input speech. These search strategies must consider a very large number of word-sequence hypotheses in a computationally efficient manner. To develop and demonstrate these techniques, we designed and implemented a complete word recognition system for continuous speech which is capable of incorporating knowledge from several sources, including lexical, phonetic, phonological, and grammatical knowledge. The complete system was given the name BYBLOS, the name of an ancient Phoenician town where the first phonetic writing was discovered. The BYBLOS system has been used as our testbed system for evaluating various speech recognition algorithms and search strategies.

In addition to developing algorithms for combining multiple knowledge sources and efficient search strategies, this project also dealt with several other issues, including: specification of the Resource Management Database and documentation of how to test with it, periodic testing to meet the agreed upon test requirements, development of several standard language models for evaluation, development of a technique for estimation of statistical language models from limited text corpora, and testing of ideas for speaker adaptation. These topics are discussed in more detail in the body of the report and in a number of papers that have been attached to this report as an Appendix.

# 2 Introduction

The most fundamental problem in speech recognition is to develop an accurate model of the acoustic signal that corresponds to any sequence of words or phonemes, in order that speech recognition can be performed. However, there are several other sources of knowledge that can be used to improve speech recognition accuracy. Some of these include: a phonetic lexicon specifying the most likely pronunciations for each word, extended by a set of phonological rules suggesting alternate pronunciations, a model of likely word sequences - based either on a heuristic model derived from rules, a statistically-based model derived by estimating probabilities from a training set, or a linguistically-based model that uses syntactic and semantic information explicitly.

The objective of this project was to develop methods and techniques to coordinate the many sources of knowledge in the decision process for a speech recognition system. This effort included developing methods for effectively combining information from the various knowledge sources, and methods for recognition search strategies that efficiently consider the tremendous number of hypotheses in the search space. To develop and demonstrate these techniques, we designed and implemented a word recognition system for continuous speech input that employed several knowledge sources. The system, which we called BYBLOS, was then used as a testbed system for evaluating various recognition algorithms and search strategies. Much of the testing of the system used the DARPA Resource Management Task, which was taken from the Navy battle management (FCCBMP) domain.

The system that was developed was based on the continuous speech phonetic recognition algorithm that had been developed in our program of basic research in continuous speech recognition for DARPA. In that work, the model for each word is derived from a set of pronunciations from a dictionary, a set of phonological rules, and from data taken from natural continuous speech. Each phonetic unit within a word is represented by a combination of a context-independent model and several context-dependent models of that phoneme. The training algorithm that was developed does not require that any speech be labeled manually. The training data only needs to be transcribed with a list of the words spoken, thus greatly reducing the amount of labor required and increasing the amount of data that can be made available for training.

In addition to developing algorithms for combining multiple knowledge sources and efficient search strategies, this project also dealt with several other issues. This included: implementing a basic testbed system for evaluating different word recognition algorithms, specification of the Resource Management Database and documentation of how to test with it, periodic testing to meet the agreed upon test requirements, and testing of ideas for speaker adaptation.

The chapters of this report are organized by topic, as follows:

The BYBLOS System

Lexical and Phonological Knowledge

Language Models

Search Strategies

System Implementation

Database Specification and Documentation

Testing System Performance and Demonstrations

Speaker Adaptation.

In each of the chapters, we recount the major areas of research under the topic of that chapter. Where applicable, we also review the major technical principles involved. Further details can be found in the set of papers included in the Appendix at the end of the report.

# 3 The BYBLOS System

Figure 1 is a block diagram of the BBN BYBLOS continuous speech recognition system. We show the different modules and knowledge sources (KS) that comprise the complete system, the arrows indicating the flow of module/KS interactions. The modules are represented by rectangular boxes. They are, starting from the top: Trainer, Word Model Generator, and Decoder. Also shown are the knowledge sources, which are represented by the elipses. They include: Acoustic-Phonetic, Lexical, and Grammatic knowledge sources. We describe briefly the various modules and how they interact with the various KSs. Additional information is given in the body of the report and in the Appendix.

### Acoustic-Phonetic Knowledge Source

The Trainer module is used for the acquisition of the acoustic-phonetic knowledge source. It takes as input a phonetic dictionary, speech to be used for training and the corresponding text transcription, and produces a database of context-dependent hidden Markov models of phonemes.

### Lexical Knowledge Source

The Word Model Generator module takes as input the phonetic models database and compiles word phonetic models, using the dictionary as another input. The dictionary is the lexical knowledge source, in which phonological rules of English are used to represent each lexical item in terms of their most likely phonetic spellings. The lexical KS imposes phonotactic contraints by allowing only legal sequences of phonemes to be hypothesized in the recognizer, reducing the search space and improving performance. The output of the Word Model Generator is a database of word models used in the recognizer.

### Grammatical Knowledge Source

In much of our speech recognition work, we use a statistical language model to represent grammatical constraints. Such models allow all word possibilities but with different probabilities such that the perplexity of the grammar is substantially lower than the size of the vocabulary. The recognition search process then uses the word phonetic models and the statistical grammar to find the most likely sequence of words, given the input speech.

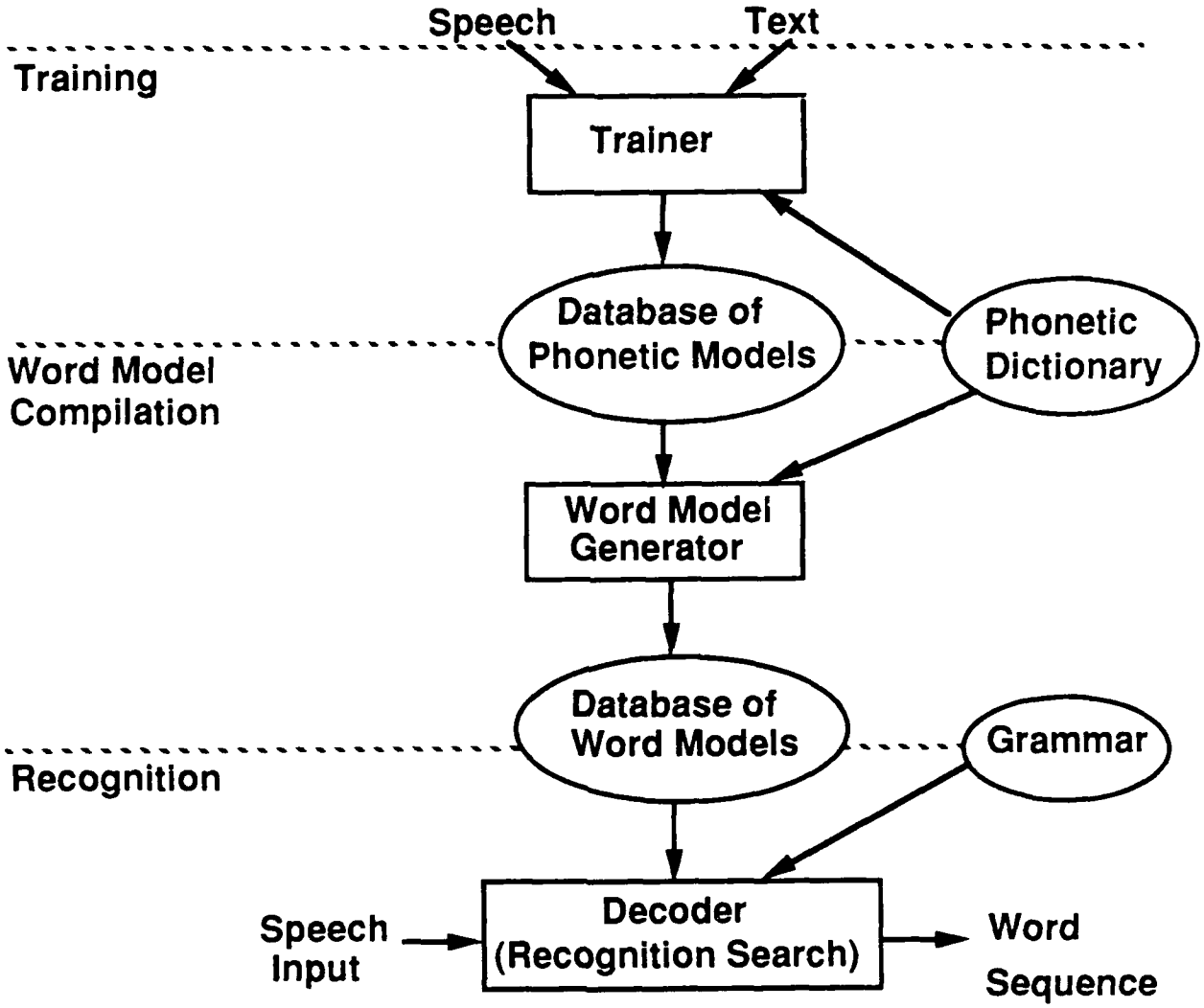



Figure 1: A block diagram of the BBN BYBLOS continuous speech recognition system.

# 4 Lexical and Phonological Knowledge

One of the basic tasks in a recognition system is to develop a phonetic dictionary (the lexicon) and to allow for the incorporation of phonological knowledge. Therefore, we developed an interactive tool that allowed a user to create and edit phonetic spellings in a dictionary. The tool could display the phonetic network corresponding to the pronunciations in a word. In addition, the system allowed phonological rules to be entered, which would then create new expanded pronunciations for a vocabulary. The system kept track of the history of each word, so that it was possible to determine which rules created which parts of any pronunciation. We used these tools to create phonetic dictionaries for the different tasks – primarily the 1000-word FCCBMP task.

One consideration that must be made in a system is the number of alternate pronunciations that should be used for each word. In principle, one would like to represent all of the likely pronunciations, since they would result in different acoustic realizations of the words. However, as we add pronunciations to a word, the difference between this word and other words decreases. In particular, if we add extra pronunciations to account for inadequacies of the phonetic recognition system, then we must constantly change this set of rules as the system improves. Furthermore, if we allow alternate pronunciations, we must be careful to represent the fact that some pronunciations are much more likely than others.

We performed experiments to determine the effect of having different numbers of pronunciations. We found, to our surprise, that the best recognition performance was achieved when we limited the number of pronunciations to one for each word. That is, even when we had a limited set of phonological rules, resulting in about two pronunciations per word, the performance was worse. On reflection, it made sense that the BYBLOS system would perform better with a single pronunciation for each word. Most of the phonological variations take place based on the context of the preceding and following phonemes. However, the system already modeled the detailed acoustic variation in phonemes with the use of context-dependent phonetic HMM models. Thus this probabilistic model for fine acoustic differences was superior to a gross phonological model of the shift from one phoneme to another. Some systems within the DARPA community had been using a large number of pronunciations – perhaps 10 or more per word. In particular, the CMU Sphinx system performance improved dramatically when all the alternate pronunciations were removed at our suggestion.

# 5 Language Models

Our first experiments with the effect of grammatical constraints on recognition performance involved changing the vocabulary size. We did this in order not to be affected by the particular words that could follow each other in a grammar. Rather we performed a set of experiments that we called "Branching Factor Experiments", in which we varied the vocabulary size in a systematic way. Given a desired vocabulary size, the recognition program would read in each sentence to be recognized. It then limited the vocabulary to the words actually in the sentence, plus enough other words to make up the desired vocabulary size. Many different random subsets of the vocabulary were chosen in order to remove any bias. This made it possible to plot the expected word recognition error rate as a function of the branching factor. We found that the error rate was typically proportional to the square root of the branching factor. We found, when we ran recognition experiments using a deterministic grammar, that this relation still held. That is, the word recognition error rate was generally proportional to the grammar perplexity.

After the basic branching factor experiments, we implemented a recognition program that would allow the recognition to be constrained by finite-state grammars. This also required building some tools that made it possible to create and manipulate grammars. We found that it was simplest to specify grammars in terms of context-free production rules. These rules were then expanded into a finite-state network. This was possible because we did not include any rules that caused recursion. The recognition program allowed deterministic finite-state automata (DFA), in which all the words leaving any state of the grammar were unique, and nondeterministic automata (NFA), in which there could be duplicate words or "null arcs" that allowed a transition from one state to another without going through a word. In general the same language can be represented with a much smaller NFA than DFA.

The first grammar that we constructed for the 1000-word FCCBMP corpus was based on the sentence patterns used to make up the sentences in the corpus. This grammar had a perplexity of only 10. We found that the sentence patterns that were used to generate the 2850 sentences in the Resource Management corpus were not very robust. That is, if a person generally familiar with the domain made up sentences using the same vocabulary, there was a high likelihood that the sentence could not have been generated by these patterns. One important issue in speech and language recognition is how to make a grammar that will cover a large percentage of new sentences.

We developed a semiautomated tool for inferring grammatical structure from a relatively small set of patterns. The tool found similarities in sentences, and then allowed that part of both sentences to be represented by a context free rule. In this way, the system generalizes the sequences of words and word classes into a hierarchical set of rules that would cover a much larger percentage of new sentences than the original sentence patterns. We applied this tool to the training subset of the sentence patterns to create a

generalized context free grammar. We compared the perplexity and coverage of new data of this new grammar to the original pattern grammar. When the test patterns were parsed using only the training patterns, none of them parsed. However, when the generalized context free grammar was used, 65% of the test patterns parsed. The perplexity of this grammar also necessarily increased, from 10 to 75.

To use this new grammar we built a version of the decoder that could use context free grammars rather than just finite state grammars. This involved first converting the set of context free rules into a recursive transition network, in which the several right hand sides for a rule are merged into a single network of terminals and nonterminals. The decoder was changed so that instead of dealing with a single finite state network of terminals (words), it could also deal with networks of nonterminals. When the symbol in a network was a terminal, it simply created a new instance of that word to be matched. When it was a nonterminal, it "pushed" down to the network corresponding to the right hand sides of that nonterminal. Since the grammar was now much larger and more interconnected, the decoder had to be optimized in several ways. One of the major optimizations was to use the forward-backward search strategy described below. This meant that when the decoder came upon any new word or nonterminal, it would know (from the forward pass) whether this word, or any of the words implied by this nonterminal could possibly be in the input speech starting at this frame.

Any deterministic grammar derived from a set of rules will have a problem in that new test data may not be able to be parsed by the grammar. For example, when new sentences were recorded at NOSC (the TONE database), we found that most of the sentences were not covered by the sentence pattern grammar or the word pair grammar. Even the generalized context free grammar covered only about 75% of the sentences that used the same vocabulary. A statistical grammar that models the probability of the next word (or word class) given the preceding words can avoid this problem by assuming that all words are possible, even though some are much more likely than others. The statistical grammar also has the additional advantage that it can accurately represent the fact that some words or classes are more likely than others. This additional information greatly reduces perplexity and increases performance. Therefore we began an effort to estimate and use a robust statistical grammar.

In the past work on statistical language models, the training set for estimating the probabilities of word sequences needed to be quite large. For example, IBM currently uses a text database of about 250 million words to estimate trigram probabilities in their office correspondence task. In many spoken language applications there is no possibility of collecting such large amounts of speech because the application does not yet exist. Rather, it may only be possible to collect on the order of 1000 sentences from a simulation of the system. To alleviate this problem we have extended the statistical grammars typically used by the use of linguistic knowledge. In particular, we group the different words in the vocabulary into classes, under the assumption that their statistics will be

relatively similar. For example, in the FCCBMP domain, one would initially assume that the names of all ships would be equally likely in any particular sentence. Therefore, when we see a training sentence that contains one ship name, we assume that we have seen a similar sentence with every other ship name. In addition, there are sequences of words that behave as a unit. For example, there are many ways of expressing a date. We can assume that in the model for a sentence, we need not distinguish among these different forms of the date. Therefore, the whole date, which may consist of several words, is treated as a single nonterminal. This greatly reduces the amount of training script that is needed. It also increases the number of words over which the grammar has effect. For example, we can now predict the probability of a particular word given that it was preceded by a preposition, followed by a date.

We developed a statistical tool that allowed us to estimate a variable order Markov chain for the sequence of word classes and nonterminals. The variable order chain has the advantage that in some contexts, there is enough training to estimate high order statistics, while in others, only first order statistics can be reliably estimated. The perplexity of this model as measured on the training data was about 20, which is very low. When measured on independent test data, the perplexity rose to about 60. However, unlike the Word-Pair grammar, which also has perplexity 60, this model would be able to recognize sentences that do not come from the grammar training set. We compared the recognition performance of this grammar with that of the Word-Pair grammar. When independent test data was used the error rates were 10% and 22%, respectively, for the two grammars. The larger error rate for the Word-Pair grammar stems in part from the fact that several word pairs in the test set were not allowed by the grammar.

This work on statistical language modeling from small corpora will be important in future work on spoken language recognition because of the need to improve recognition performance through the use of statistics, and because the training sets for new tasks will always be small.

# 6 Search Strategies

In this chapter we describe our work on developing efficient search strategies for finding the most likely word sequence given the acoustic signal. In principle, the best algorithm for determining the word sequence given an acoustic signal is to consider all possible word strings exhaustively. For each word string we must compute a score (conditional probability) of that word sequence, taking into account all the sources of knowledge available. Then we simply choose the word sequence with the highest score as our answer. This algorithm, which we would call a tightly coupled, top-down search, guarantees the minimum error rate for a given set of knowledge sources. However, this exhaustive search is clearly infeasible. Therefore, we must develop search strategies that approximate this algorithm, with computation that is acceptable. However, throughout our work in developing efficient search strategies, we always must keep in mind that we are trying to approximate the effect of this exhaustive search. The remainder of this chapter enumerates several of the different search search strategies that we have developed under this project.

Some of the general principles that were established for a desirable search strategy were:

1. Use the computationally "inexpensive" knowledge sources to reduce the number of choices drastically, and then use the more expensive knowledge sources on this reduced set.
2. Any decisions made in (1) must be made in a way that almost never makes a mistake, otherwise these "irrecoverable errors" will multiply and dominate the errors.

Two search strategies that are commonly used in speech recognition are the Best-First stack search, and the Viterbi beam search. The best-first search considers only the best theory at a particular time. It extends this best theory by all possible next words, scores all these new theories using all available sources of knowledge, and reinserts the new scored theories back into a stack that is sorted by theory score. This algorithm has the theoretical advantage that, if the scores are meaningful, it should do the least amount of computation. However, in practice, it is very difficult to sort the theories appropriately. In particular, it is very hard to compare two theories that span different regions of the input speech. Therefore, even with the many heuristics that are used, this strategy often finds a suboptimal answer, or results in tremendous amounts of computation.

The Viterbi beam search is much easier to implement than the best-first search and has many desirable properties. First, it guarantees to find the sequence of states of a finite-state hidden Markov model with computation that is proportional to the number of states and the length of the input speech. The beam search implies that at each time frame, all theories that have a probability that is sufficiently far below the probability of

the most likely theory are removed, since they are not likely to result in a better score by the end of the utterance.

Unfortunately, there are still two problems with the Viterbi search algorithm. First, the algorithm finds the most likely sequence of states, rather than the most likely sequence of words. While this difference is often small, it does introduce some unnecessary errors. Second, and more serious, most interesting models of language have a very large number (if not infinite) number of states. For example, even a context-free language cannot be represented using a finite number of states. Therefore the computation associated with the straightforward beam search is often excessive.

In 1985 we devised an algorithm that has the simplicity of the Viterbi algorithm but computes a score that more closely approximates the "true" score of the most likely sequence of words. Stated simply, the algorithm is just like the Viterbi algorithm, except that at each state, where the Viterbi algorithm keeps the *maximum* score from all preceding states, our algorithm *adds* the scores from all the preceding states. We call this algorithm a *pseudo-Baum-Welch* search for the most likely word sequence. In several experiments we verified that this algorithm results in somewhat lower error rates than the Viterbi algorithm. The interesting observation was that the difference in error rate was relatively constant over different applications. That is, our pseudo-Baum-Welch algorithm consistently resulted in 2% fewer errors than the Viterbi algorithm, whether the original error rate was 30% or 5%. Therefore, when the error rate was low (which it must be for a useful system), the difference was important.

There were certain remaining problems with the algorithm described above. Since the score produced by this algorithm is not exactly the same as the true score, there is still some chance that it will not find the word sequence that has the highest true score. In addition, since we use a pruning algorithm to try to avoid computing most of the scores, it is possible that the algorithm eliminates the correct word sequence from consideration without computing its full score. Because of these two problems it is helpful to know how the true score of the correct answer compares with the true score of any incorrect answer that the recognition program finds. Therefore we added a feature to the decoder that allows the system to find the true score for any particular word sequence for a sentence, by scoring only that word sequence. We call this a *forced scoring algorithm*. Whenever the decoder finds the wrong word sequence, this forced scoring algorithm can then be used to find the true score for the correct word sequence and the true score for the word sequence that the decoder found. If the true score for the correct word sequence is higher than the score for the word sequence found, then we know that it was due either to the pseudo-Baum-Welch score being different from the true score, or due to pruning out the correct answer. When we tested our search algorithm using this new feature, we found that whenever the decoder finds an incorrect answer, the incorrect answer always has a higher true score than the correct answer. This confirms that the decoding algorithm is empirically optimal.

As described above, it is also possible that search errors are the result of pruning out the correct answer because, at some point in the utterance, it was scoring much worse than some other hypothesis. Unfortunately, unless the pruning is very conservative, there is some small probability that the most likely sequence of words is pruned out from consideration due to a small region where it scores poorly. While this might happen only once in 25 sentences, it represents an unwanted noise in our estimation of error rates. To alleviate this problem for research runs, we use the heuristic described above to detect errors due to pruning and rerun the sentence with more conservative pruning if necessary. The utterance is first run at a very aggressive pruning level, which results in a factor of 10 speed up. If the answer found is incorrect, the sentence is run again forcing the correct answer, and then the answer that was found during the search. Both of these require very little time, since only one sequence of words is possible. If the score for the correct answer is actually higher than that for the incorrect answer, then the utterance is rerun with a very conservative pruning threshold. We find that, including the utterances that need to be rerun, the net effect is a factor of about 5 in the speed of research runs of the decoder. Of course, when the decoder is running in any real application or formal evaluation, it doesn't know the correct answer, and so it must use a more conservative pruning or accept some increased error rate.

The more serious problem mentioned at the beginning of this chapter was that for large language models, the number of word states that need to be scored in each frame can often be significantly larger than the number of words in the vocabulary. For example, the Sentence Pattern Grammar, which is a large finite state grammar, has about 100,000 arcs initially and about 30,000 arcs in its most compressed form. Therefore, if the pruning were not able to eliminate more than 97% of the words from consideration, the number of active words in the beam search would be larger than the 1000-word vocabulary. Therefore, we have developed a new class of recognition search strategies, which we call *multiple-pass* search strategies, that is useful for speeding up the search with large grammars, such as statistical grammars and natural language grammars. These algorithms find upperbound scores for each of the words in the vocabulary in different regions of the input. Then, while performing a grammar-directed acoustic search, the decoder considers only those words that are known to be likely given the input speech. The particular version of this paradigm that we implemented has been named the *forward-backward* search because of its similarity to the forward-backward training algorithm.

As the syntax-directed search is proceeding left-to-right through an utterance, it must extend each theory in which a word has ended by all the possible following words. The beam search reduces the number of theories by eliminating those for which the sentence so far scores badly, compared to the other theories. The beam search would be much more effective if, at this point in the utterance, it could know the score that the remainder of the utterance would receive also. Then the pruning could be based on this total score. Of course, computing this score is equivalent to performing a full decode, which is what we are trying to avoid. However, say we knew the score for the most likely

sequence of words from this point in the utterance to the end of the utterance (ignoring grammatical constraints). This score would be an upper bound on the actual score that would be found were it computed exactly. Furthermore, say we knew this upperbound score for each possible first word in the remaining string. Now, we could consider each of the words that can come next, according to the grammar, and for each one, look up the upperbound score of a sequence of words beginning with that word at this point in the utterance. This score, when added to the score of the theory to the left, can then be used in the beam pruning, thus eliminating most of the possible continuations of this theory. The only problem with this algorithm is that we haven't seen the rest of the utterance yet (assuming that the algorithm is running in close to real time), so we cannot possibly compute the scores of word sequences in the future. However, if we turn the problem around, there is a solution that is feasible. Let us say that as the speech is given to the decoder (in real time), it computes the scores as if it were using no grammar in the recognition. At each frame, it remembers the score of word sequences ending with each possible word in the vocabulary. Only a small fraction of the words in the vocabulary will end with a good score at each frame. When the end of the utterance is detected, the decoder then begins a grammar-directed search, but in the reverse direction. This time, since most words have been eliminated, the decoding proceeds much faster than real time. The most likely answer is then found with only a short delay past the end of the utterance. With a small modification, this algorithm can also be made to run forward, in order to eliminate even the small delay at the end of the utterance.

We have used the forward-backward search algorithm described above to speed up the search for several very large grammars. These include the Sentence-Pattern grammar, a high-order statistical grammar, and a recursive transition network grammar. Our experiments indicate that, for these large grammars, the increase in speed is at least a factor of ten, when we use the forward-backward search algorithm. A modification of this algorithm would use a first-order statistical grammar in the first pass, in order to reduce the choices further.

# 7 System Implementation

Much of the work of this project was necessarily spent implementing the different training and recognition algorithms. The initial set of algorithms for word-based training and recognition were implemented on the Symbolics Lisp machine using Zeta-Lisp. The flexible environment made it relatively easy to implement several algorithms quickly. However, the resulting programs were not very fast, since Lisp tends to result in slow computation. The result was that many experiments were impractical to run, since the time that they required was too great. In particular, it was frequently more than the mean time between failure of the machines.

More recently, we have completely redesigned and reimplemented all of the algorithms in C on the SUN4 workstations. The implementation takes somewhat longer, but the resulting programs run about an order of magnitude faster. As a result, it is possible to run several different versions of the programs and to tune different parameters. One of the direct consequences has been a marked improvement in the recognition accuracy of the system. In addition, since the language used is more portable, we will be able to take advantages of newer, faster machines as they become available, without having to redesign all the algorithms. We are currently investigating several faster machines that would increase our computing power by at least a factor of five.

# 8 Database Specification and Documentation

One of the major contributions of the current DARPA program in speech recognition has been the specification, recording, and adoption of a standard corpus of sentences for research and standard testing. BBN has been instrumental in the specification of the corpus and the testing standards to be used with that corpus.

Several speech corpora were defined to serve the different research needs of the community. We felt that it would be important to have a wide range in the amount of training speech available for each speaker, as well as a very large number of speakers available. Therefore the basic makeup of the corpus was designed so that there would be different sections. The first would contain 640 speakers, each saying 10 sentences. The second would contain 160 speakers, each saying 40 sentences. The third would be geared for speaker-dependent research, and would contain 12 speakers with 600 training sentences for each. Finally, there would be 2 to 4 speakers with 2 to 4 hours of speech each. The 640 speaker corpus consisted of material designed for detailed phonetic research, and was thus phonetically marked. This corpus has been called the TIMIT database. The other corpora consisted of sentences pertaining to the Resource Management task domain. Each of the corpora contained designated training sets, development test sets to be used while trying out new algorithms, and evaluation test sets for formal testing.

We specified the sentences in the FCCBMP battle management domain (later to be called the Resource Management task domain) through lengthy discussions with people at NOSC. This task involved becoming familiar enough with the application and likely uses to generate a 1000-word vocabulary and about 1000 different sentences with database queries, display commands, and expert system questions. As such, the task combined the domains that reside in several different systems, most notably, IDB, OSGP, and FRESH. After the initial sentences had been composed, and checked by NOSC, they were converted into sentence patterns by replacing the open class words by their classes. Then, as many sentences as desired could be generated. We generated three sentences from each pattern, resulting in approximately 2850 sentences. These sentences were then sent to TI, where some additional changes were made before recording. (Some words that were too hard to pronounce were replaced with other words.) The procedures that were followed in creating this extensive corpus were documented in an ICASSP paper, which is included as an appendix to this report.

In order to be able to compare recognition results accross different research sites using different algorithms, it was necessary to assure that all sites were using the same grammatical constraints. Since the BYBLOS system was the first one within the program to produce reasonable recognition results, we inherited the task of specifying and trying standard test conditions. We documented the phonetic dictionary that we had developed and made it available to other sites. We showed that it was not advantageous to have a large number of phonetic pronunciations for each word, since this made the different

words more similar. Finally, we provided documentation for three standard test grammars:

1. No grammar - perplexity 1000.
2. Sentence Pattern Grammar - perplexity 10
3. Word-Pair Grammar - perplexity 60

The first grammar used for testing was the null grammar that allowed any sequence of words. This grammar tested the basic word recognition capabilities of the systems. The sentence pattern grammar was a grammar derived from the patterns used to generate the sentences in the Resource Management corpus. Since we knew that this grammar was unrealistically constrained, we created another grammar that allowed all pairs of words that could occur anywhere in the sentence pattern grammar. This increased the perplexity to 60, making it a much more reasonable grammar. By using these three grammars, it was possible to evaluate the word recognition capabilities of each of the sites at different levels of difficulty.

One way of estimating the difficulty of a task is to measure the average number of words that can come after each word in the language model used with the task. The mathematical quantity that we use for this is called perplexity. While this measure doesn't take into account the phonetic similarity between words, it has been found to correlate well with word recogition error rate. We wrote a technical note to document the precise techniques used to measure the perlexity of a language model on any particular test set of sentences. That note is included as an appendix to this report.

Finally, as the word recognition capabilities of the different systems has improved, there has been a need for a grammar that is more difficult than the Word-Pair grammar. While the recognition performance is not that high that it would be useful, the number of errors in a reasonable-sized test set is not large enough to be measured with statistical reliability. Furthermore, testing with no grammar is also too unrealistic, because it requires many distinctions that would never be needed in a system. Therefore, we developed a new standard test grammar based on a first order statistical model of word classes. The grammar, which is based on only 100 word classes, was designed to have a statistical perplexity of about 100, which will result in about twice the error rate associated with the word-pair grammar. We estimate that the difficulty of this grammar is comparable to the difficulty of a more realistic task with about 5000 words, where the test (actual) sentences may not be quite so similar to the training sentences. We documented and distributed a set of programs for estimating and constructing this grammar from an annotated lexicon and a corpus of sentences that has only orthographic transcriptions.

# 9 Testing System Performance and Demonstrations

Throughout the project we have performed several formal and informal evaluations of the recognition accuracy of the BYBLOS system. The early tests and demonstrations were informal, since BYBLOS was the only complete system at that time. The later tests were more formal. The formal tests have been in accordance with rules agreed upon among the different research sites, together with NBS (now NIST).

Our early work in context-dependent phonetic hidden Markov models for phonetic recognition was incorporated into a word recognition system during 1985. The first test of this system was on a small (334 words) electronic mail task. The domain consisted of commands to an electronic mail system. 300 training sentences and 100 test sentences were recorded from each of 3 male speakers. We compared the recognition accuracy with context-dependent phonetic models of different types with the accuracy when context-independent models were used. The recognition experiments were run first with no grammar, and later with a finite-state grammar made up to model all of the sentences. Averaged over the 3 speakers, the word recognition accuracy with context-independent models was 76%. When context-dependent models were used the accuracy was 90%. When a grammar with perplexity 31 was used, the recognition accuracy improved from 94% to 98.2%. This represented convincing proof of the viability of the use of context-dependent phonetic models and of the use of HMM models in general.

During the next several months of 1986 we implemented a 350-word subset of the FCCBMP Resource Management Task Domain. This involved recording training and test sentences for the new domain and running similar tests to those described above. The results were quite similar. Next, we added 300 new words to the test vocabulary, to test the effect of having test words that were not included in the training. Most of the words added were additional names of ships and ports. We found that the recognition results were quite similar to those reported earlier.

During the spring of 1986 we implemented a demonstration of the BYBLOS system that would allow a user to speak a sentence and have the answer appear about one minute later. The system, which ran on a Symbolics Lisp machine displayed its progress as it attempted to recognize what was said. In particular, it displayed a tree of the most likely sentence hypotheses that were under consideration. In July, 1986 we held a demonstration of this system at BBN.

In addition to using speaker-dependent models derived from 300 sentences from one speaker, we demonstrated the speaker adaptation capability of the system. Forty sentences were recorded from each of the visitors. These sentences were used to transform a speaker-dependent model from one speaker so that it could be used for the new speaker. While the recognition accuracy with the adapted model was not as high as for the speaker-dependent model, it was still quite reasonable.

During the end of 1986 we tested the BYBLOS system on the full 1000-word vocabulary. The tests were run on two speakers from BBN, since the data being recorded at TI was not yet available. The word recognition accuracy with no grammar was 87%.

The speech data for four speakers was made available in January of 1987. This was the first formal test of the system using data from outside BBN. In this case we had 600 sentences (approximately 30 minutes of speech) from each speaker. We ran tests under three grammar conditions:

1. Sentence Pattern Grammar - perplexity 10
2. Word-Pair Grammar - perplexity 60
3. No grammar - perplexity 1000.

The recognition results are presented for each of the grammars and for the four speakers from TI as well as the two speakers from BBN.

| | Sentence Pattern | Word Pair | No Grammar |
|---|---|---|---|
| TI-4spkrs | 97.8% | 89.9% | 65% |
| BBN-2spkrs | 99.8% | 98.2% | 87% |

The results showed clearly that the recognition accuracy depends significantly on the grammar used. They also indicated that the speech recorded at BBN resulted in much higher recognition accuracy than that recorded at TI. This was presumably due – at least partially – to the speakers at BBN speaking more carefully. It also indicates that a significant improvement in system performance can be achieved by a certain amount of instruction to prospective users in how to use the system. This improvement is at least comparable to the difference in performance resulting from algorithm improvements.

On July 27, 1987 we gave a demonstration that showed how an integrated spoken language system could be used for the FCCBMP application. A graphics system that enabled a user to manipulate objects on a map was connected to the natural language system, so that typed commands and questions would be answered and would result in appropriate displays on the map. The output of the speech recognition was then connected to the natural language so that commands and questions could be spoken. While the connection between the speech and the natural language was serial, it illustrated the power that such a spoken language system would have.

One of the requirements for the October, 1987 meeting was that some of the results reported would be from a "live test", which meant that the speakers were speaking directly to the system, which would recognize each sentence and display the answer before they would speak the next sentence. On July 27, the three speakers who were to be in the test came to BBN to provide training speech in order that we could compute speaker-dependent models for the speakers. (The speakers were Alan Sears, David Pallett, and

Tice DeYoung.) Each speaker read training sentences during a total elapsed time that was limited to one hour. The recording took place in two half-hour sessions. Afterwards, we listened to all of the recorded sentences and deleted those where the words spoken were different from those in the text transcriptions. On the average, 80% of the utterances were kept, resulting in about 350 training utterances for each speaker, or about 18 minutes of actual speech. On September 29, the three speakers returned to test the system. The word models for each of the speakers were transfered to the Butterfly computer which performed the recognition. The grammar used was the Word-Pair Grammar. Each of the speakers read 30 test sentences, one by one, and waited for the recognition answer to be typed out. All input data and recognition results were also saved on files for later analysis. On average, the recognition time was 10-40 seconds, or about 10 times real time. In each case, the speaker was able to finish the entire session (including putting on the microphone, comments, adjusting levels, and false starts) within 1/2 hour of elapsed time. The word recogition error rates for the three speakers were AS: 4.4%, DP: 5%, TD: 12%. These same sentences were also processed on the Lisp Machine simulation of the decoder to provide recognition results with no grammar.

In addition to the live tests, there were also formal tests run using the data recorded at TI. In this case, test data from eight of the speakers was evaluated. Four of the speakers had been used in the tests performed in March, 1987. The speaker-dependent models were generated using 570 of the 600 training sentences (we reserved 30 for in-house testing). In August we received from NBS the set of 25 test sentences to be used for testing. Again, tests were run using both the Word-Pair Grammar and no grammar. We no longer used the Sentence Pattern Grammar, since it was judged to be unrealistically easy. The results were consistent with those obtained in March. The average error rates were 32% with no grammar, and 7.5% with the Word-Pair Grammar.

During the October, 1987 meeting we demonstrated the BYBLOS system in the conference room where the meeting was held. There were several technical problems related to getting the audio signal from this room back to our A/D facility, which was several hundred yards away in a different building. The solution that was finally chosen was to transmit the signal over unused telephone wires that went between the buildings. The demonstrations included a near-real time demonstration of recognition on the Butterfly Parallel Processing system. In this demonstration, the system displayed the hypothesized word string as it processed the sentence. Frequently, the first two or three words were displayed before the speaker finished speaking the sentence. Several speakers were used during this demonstration.

In April, 1988 we tested the new speaker adaptation algorithms developed under the Basic Research effort on the Resource Management Database, and the data collected "live" at BBN. We found that, on the average, the performance of the system when models were adapted using two minutes of speech from the new speaker was equal to that derived when 18 minutes of speech from the new speaker was used with the speaker-

dependent training algorithm. This is a significant improvement over our previous speaker adaptation algorithm, which resulted in performance equivalent to about 8 minutes of speaker-dependent training.

During early 1988, we redesigned and implemented the BYBLOS system on the SUN4 workstations. Previously, on the Symbolics Lisp machine, the limitations of slow computation and limited virtual address space made it difficult to run many experiments. In particular, we were not able to use multiple sets of spectral parameters in the training and recognition. The primary difference in the SUN4 version was that we now used the derivatives of the cepstral parameters in addition to the cepstral parameters themselves. In addition, it was now possible to run many experiments in order to tune various system parameters. All tuning was done on parts of the training set or on the October, 1987 test set. We then ran the May 1988 test data for all 12 speakers through the decoder using the word-pair grammar and the null grammar; the word error rate was now 3.4% and 16.2% respectively. This represented a word error rate reduction by a factor of two relative to the previous system.

Shortly after running these tests, we completed our work on smoothing the probability distributions of the HMMs. When we received the test data for the February 1989 meeting, we decided to run the experiments both with and without the smoothing algorithm. At the meeting in February we presented the effect of smoothing on both the May '88 and February '89 test sets. The word error rates are given below.

| | Word-Pair | | No Grammar | |
|---|---|---|---|---|
| | Control | Smoothing | Control | Smoothing |
| May '88 | 3.4 | 2.7 | 16.2 | 15.2 |
| Feb '89 | 2.9 | 3.1 | 15.3 | 13.8 |

This showed that, although the smoothing algorithm helped in most cases, it did not always improve the performance. However, the overall results were the best reported to date by any other research site on this corpus.

# 10 Speaker Adaptation

A key area of our work has been in rapid speaker adaptation. Most of the systems reported to date are based on two training paradigms. In the speaker-dependent paradigm, a relatively large amount of speech from the user is collected, in order to estimate a very accurate model of how that speaker speaks. The result is very high recognition accuracy. In the speaker-independent paradigm, speech is collected from a very large number of speakers (at least 100 speakers). This speech is pooled as if it all came from a single speaker, and is used with exactly the *same* algorithm as in the speaker-dependent case. The result is that when a new speaker speaks to the system, the recognition accuracy is not degenerately bad, but the word error rate is still about a factor of 2 to 3 times that in the well-trained speaker-dependent paradigm. In addition, for many applications, it would be impractical to collect speech from a large number of speakers just for that application.

The speaker adaptation paradigm provides an alternative to these two approaches. The algorithm, which is quite different from the basic speaker-dependent/independent algorithm, starts with a well-trained model from a single reference speaker. This speaker is presumably one who has trained the system in the speaker-dependent paradigm. Then, a small amount of speech is collected from the new (target) speaker. This speech is used to transform all of the models of the reference speaker so that they are appropriate for the target speaker. The resulting system performance is somewhat better than that in the speaker-independent paradigm, but somewhat worse than the speaker-dependent paradigm, at a small fraction of the cost of data collection. This paradigm has the additional advantage that it will be natural for the user to adapt the system whenever the acoustic environment or his voice should change for any reason.

We have investigated several new algorithms for rapid speaker adaptation. The major contribution of this effort has been a probabilistic spectral mapping algorithm that transforms the reference speaker model into a target speaker model. We have experimented with several algorithms for estimating this mapping, and have presented recognition performance results at several of the project meetings.

This area remains one of our key research areas, since we feel that ultimately the rapid adaptation paradigm will be the most practical for new users. The system will begin by prompting a new user to read a small number of sentences. Then, as the user speaks to the system, it will incrementally improve the performance until it eventually becomes a high-performing speaker-dependent system.

# Appendix

This appendix contains a number of papers that have been written under this contract. Below is the list of papers included.

1. O.Kimball, P. Price, S. Roucos, R. Schwartz, F. Kubala, Y.-L. Chow, A. Haas, M. Krasner, J. Makhoul, Recognition performance and grammatical constraints, Proc. DARPA Workshop on Continous Speech Recognition, Palo Alto, CA, February 1986.

2. Y.L. Chow, M.O. Dunham, O.A. Kimball, M.A. Krasner, G.F. Kubala, J. Makhoul, P.J. Price, S. Roucos, and R.M. Schwartz, BYBLOS: The BBN continuous speech recognition system, IEEE International Conference on Acoustics, Speech, and Signal Processing, Dallas, Texas, pp.89-93, April 1987.

3. P. Price, W.M. Fisher, J. Bernstein, and D.S. Pallett, The DARPA 1000-word resource management database for continuous speech recognition, IEEE International Conference on Acoustics, Speech, and Signal Processing, New York, N.Y., pp. 651-654, April 1988.

4. J.R. Rohlicek, Y.-L. Chow, and S. Roucos, Statistical language modeling using a small corpus from an application domain, IEEE International Conference on Acoustics, Speech, and Signal Processing, New York, N.Y., pp. 267-270, April 1988.

5. F. Kubala, Y. Chow, A. Derr, M. Feng, O. Kimball, J. Makhoul, P. Price, J. Rohlicek, S. Roucos, R. Schwartz, and J. Vandegrift, Continuous speech recognition results of the BYBLOS system on the DARPA 1000-word resource management database, IEEE International Conference on Acoustics, Speech, and Signal Processing, New York, N.Y., pp. 291-294, April 1988.

6. S. Roucos, Measuring perplexity of language models used in speech recognizers, Paper sent to DARPA sites, 1988.

7. M.-W. Feng, Iterative normalization for speaker-adaptive training in continuous speech recognition. IEEE International Conference on Acoutics, Speech, and Signal Processing, Glascow, Scotland, pp. 612-615, May 1989.

Proceedings of a Workshop Held at Palo Alto, California February 19-20, 1986.
Sponsored by DARPA.

# RECOGNITION PERFORMANCE AND GRAMMATICAL CONSTRAINTS

O Kimball, P Price, S Roucos, R Schwartz, F Kubala,
Y -L Chow, A Haas, M Krasner, J Makhoul

BBN Laboratories, Inc
10 Moulton Street
Cambridge, MA 02238

**ABSTRACT**

We describe the integration of grammatical with acoustic knowledge sources in the BBN continuous word recognition system, and the resulting effects on performance. This combination decreases the total number of insertions, deletions and substitutions by a factor of more than 6 compared to the system with no grammatical constraints, and yields a word accuracy of better than 98%. We show that constraining the set of possible word sequences can improve performance, even when the amount of training per lexical item remains fixed. In addition, we address the issues of estimating from limited data the degree of constraint imposed by a grammar and the importance of incorporating acoustic similarity in such measures.[1]

## 1 INTRODUCTION

In this report we describe the development and use of various finite state grammars in the BBN continuous speech recognition system. In particular, we investigate the relationship between recognition performance and the degree of constraint imposed by a grammar. We feel that understanding such relationships is crucial to evaluating how well specific techniques of linguistic modeling can be generalized to larger and more complex tasks.

It is well known that recognition performance improves as vocabulary size decreases. Similarly, when syntactic and semantic information are used to reduce the number of words that can legally follow a given sequence of words, a recognizer is expected to make fewer errors. Two related measures of this type of grammatical constraint are perplexity and branching factor. decreasing these characteristics of a grammar should lead to improved performance. We shall discuss how these measures can be estimated when only a small set of representative sentences are available.

In the following section we describe our recognition system. In section 3, we describe a set of experiments designed to demonstrate the relationship of performance to branching factor when the amount of training per item remains constant. We then address the issue of estimating degree of grammatical constraint from limited data (section 4). In section 5, we describe the incorporation of various grammars in our recognition system and the resulting effects on performance.

## 2 THE SPEECH RECOGNITION SYSTEM

The speech recognition system consists of a feature extraction stage, an acoustic scoring and a linguistic scoring. The feature extraction stage computes the short-time spectral envelope every centisecond and represents it by 14 Mel-warped cepstral coefficients. A vector quantizer discretizes the spectral envelope to one of 256 spectral templates using Euclidean distance. The sequence of discrete spectra is used to compute the likelihoods of all possible hypotheses in the acoustic and linguistic scoring modules. Recognizing an input utterance involves finding the sequence of words $w_1 \ldots w_m$ that maximizes

$$P(x_1 x_2 \ldots x_n \mid w_1 \ldots w_m)\, P(w_1 \ldots w_m)$$

where $x_1 \ldots x_n$ is the sequence of quantized spectra and $w_1 \ldots w_m$ is a sequence of words. The first term, the acoustic score, is derived from a hidden-Markov model (HMM) for each word. The second term, the

[1]This work was sponsored by the Defense Advanced Research Projects Agency and was monitored by the Office of Naval Research under contract number N00039-85-C-0423

linguistic score, is, in principle, a model of the expected syntax and semantics. This term includes a model of duration (longer sequences are less likely), and a grammatical score. At present, due to limited data, the grammatical score is simply set to 1 for sentences allowed by the grammar and to 0 otherwise.

The dictionary used was developed and made available to us by the speech group at Carnegie-Mellon University. We expanded it (from about 200 words) to 334 words in order to fill out categories that were represented in the original version. In particular, our version includes all months, all days of the week, possessives for all proper nouns and plurals for all other nouns, and cardinals and ordinals to cover numbers up to 999.

The training for our system was on 300 sentences (about 15 minutes) for each talker. These sentences were syntactically and lexically based on 100 example sentences also provided by CMU. We reserved the set of 100 sentences for testing. The sentences were designed to be representative of human-machine interaction in an electronic mail task, referred to as the Email task.

Our word models are phonetically based and capture the acoustic coarticulatory effects within a word to the extent that they can be estimated reliably from available training data. In short, to obtain robust estimates of the transition and output distributions of the HMM for a phoneme-in-context we use a weighted average of the parameters of models with varying amount of context. The details of these word models are discussed in [2].

The linguistic model, which computes the *a priori* probability of a word sequence, uses one of two types of models for the language. The first model has no grammar and allows any word sequence. In this case, the probability of a word sequence is determined by its length.

$$P[w_1 \ldots w_k] = c\, a^{-k}$$

where a is just an insertion penalty that is chosen empirically to control the insertion rate of the recognizer output and c is a normalizing constant. The second language model is a finite state automaton. We describe in a later section how we generated the finite state grammars from a small corpus of sentences. At present, sentences are either accepted or rejected as grammatical depending on whether the automaton parses them or not. Given sufficient data to determine the likelihood of different word sequences, the paths of the automaton could be modified to impose probabilities on sentences of the grammar.

# 3 RECOGNITION ACCURACY AND BRANCHING FACTOR

It is well known that recognition performance improves with smaller vocabulary size, with or without grammatical constraints. The improved performance may stem from two factors: (1) the smaller set of elements that need to be distinguished, and (2) the greater amount of training that can be devoted to each of the items. As vocabulary size increases, comparable training becomes more difficult. Since our goals involve increasing vocabulary size, we felt it was important to establish that the first of the above factors alone, i.e., smaller vocabulary size (which can be simulated by using a grammar), is sufficient to improve performance without increasing the amount of training per lexical item. Further, we would like to investigate the relationship between performance and constraints such as vocabulary size or grammaticality. A set of experiments was designed to simulate the effect of grammatical constraints over a range of branching factors. This was done by restricting the set of lexical items to the words appearing in a given test sentence plus additional words selected randomly from the dictionary until the total number of words is equal to the desired branching factor.

## 3.1 Methodology

We investigated branching factors of 10, 20, 50, 100, 200, and 334. The last figure includes the entire dictionary. Performance was assessed for the task of recognizing 30 of the 100 test sentences, described earlier, as produced by three male talkers. Since we had previously made changes in our system based on recognition of these 30 sentences, we repeated the experiment for the smallest and largest branching factors on the 70 previously unused sentences. Since performance at these points for the new sentences did not differ greatly from the results based on the 30 sentences (performance was actually about 1% better on the new sentences), we present the results based on the 30 sentences.

In order to achieve comparable statistical significance across the tests at various branching factors (BF), that is, to adequately sample the dictionary for each, we increased the number of repetitions for experiments at lower BF. BF of 10 was repeated at least 10 times per talker per sentence, BF 20 (11 times), BF 50 (6 times), BF 100 (3 times), BF 200 (twice) and BF 334 (once).

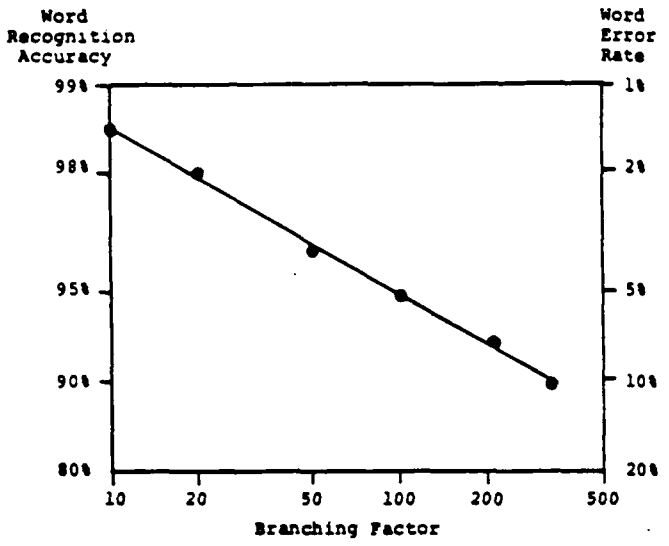



Figure 1: Performance and Branching Factor. Plotted is word accuracy, (substitutions + deletions) divided by the total number of words in the test sentences, averaged across 3 male speakers, as a function of branching factor.

### 3.2 Results and Discussion

Figure 1 shows the error rate averaged across the 3 talkers' productions of the 30 test sentences. Performance is plotted as a function of branching factor on a log-log scale. It is seen that performance increases (linearly on this scale) with smaller branching factors. word accuracy improves from about 90% for the full dictionary to about 98.5% for the branching factor of 10. As mentioned earlier, performance on the remaining 70 test sentences was about 1% better for branching factor of 10. The repetitions of the experiment allow us to sample the effects of various choices of vocabulary items, but not the effects of variability in articulation. In fact, our entire set of errors for the branching factor of 10 correspond to one or two words produced by each talker. Given this distribution of errors and the difference between the percentage of errors on the two sets of sentences, we conclude that 30 test sentences (187 words per talker) are not sufficient to reliably estimate performance in this case. The experiment has, however, confirmed our hypothesis that reduction of the number of allowable words is sufficient to improve performance without increasing training, and we feel that the methodology may prove useful for estimating the performance of a recognition algorithm on tasks differing in the complexity required of the grammar. In order to quantify this complexity, we present several methods for estimating the amount of constraint imposed by a grammar.

## 4 ESTIMATING GRAMMATICAL CONSTRAINT

When recognition is performed without a grammar, the set of possible outcomes is the set of all possible combinations of the lexical items. The role of a grammar is to disallow some of those combinations. This means that at any point the grammar has to choose not from the entire set of lexical items, but from a smaller set. By reducing the legal possibilities the grammar imposes a constraint which makes the recognizer's task easier. How does one measure the constraint imposed by the grammar? One would like to average the number of choices at various points and weight them according to how likely they are to occur. Such a measure, based on the information theoretic concept of entropy, exists and is called "perplexity" [1]. For a deterministic finite state automaton we define its entropy, H, by

$$H = \sum p(i)\, h(i)$$

where $p(i)$ is the probability of node i, and $h(i)$ is the entropy of the set of choices emanating from that node. The perplexity, Q is

$$Q = 2^H$$

The perplexity of a grammar is determined by the network connectivity and the probability assignment of the different transitions. In our case, the network connectivity is determined by the types of linguistic phenomena captured in a particular grammar. The probability assignment of the transitions is, however, more difficult. The basis for our grammar was a set of 100 sentences intended to represent rather than to define the language. In fact, many different grammars can be built to cover all or most of these sentences while differing greatly in the number and type of additional sentences covered, and, more importantly, differing in their perplexity. The problem now becomes the estimation of perplexity given a set of "representative" sentences. We propose three methods

The first is the maximum perplexity of a finite language [6] which is obtained by solving for the positive root $x_0$ of

$$\sum_{k=1}^{l_{max}} N_k x^{-k} = 1$$

where $N_k$ is the number of sentences of length $k$ in the language, $l_{max}$ is the length of the longest sentence in the language, and $x_0$ is the desired maximum perplexity

A second measure, which we will call the uniform branching estimate of perplexity, is obtained by assuming all transitions from a node in the grammar to be equally likely

The third measure, called test set branching factor, uses the set of test sentences to estimate the average branching factor encountered by traversing the FS network along the p 'hs corresponding to each sentence. We use the geometric mean of the number of branches at each node over all the test sentences as an estimate of task perplexity.

All the above measures ignore the acoustic similarity of the words, an important factor. Measures including this factor have been proposed, see, for example, [3].

# 5 RECOGNITION ACCURACY AND GRAMMATICAL CONSTRAINTS

In this section, we compare recognition performance using grammars differing in the degree to which they constrain the set of allowable word sequences. We began with a grammar designed to cover a structural subset of the Email sentences, the commands. A goal of this grammar was to maximize coverage of these sentences plus logical extensions suited to the Email task environment. Equally important in the design of this grammar was the minimization of "over-generation", i.e., the generation or acceptance of many ungrammatical sentences.

Our interest in grammars is broader than simply improving performance on a given task In addition, we would like to investigate the trade-off in performance versus over-generation, and to estimate performance on more difficult tasks, i.e., tasks requiring a larger number of choices at various points in the grammar We therefore designed a second grammar for the commands a grammar with greater perplexity Similarly, we designed two grammars differing in perplexity for the entire set of sentences (commands as well as questions)

## 5.1 Integration of Grammatical Constraints in the Recognition System

We approached the implementation of a grammar in our recognition system in two steps First we created a description of the Email task language in a modified context-free notation This description was based on the 100 sentences mentioned earlier, and was designed to capture generalizations of the linguistic phenomena found in them. Second, we created tools that transformed this description into structures in our recognizer that provide the corresponding grammatical constraint. These tools provide us with a general facility for capturing in our recognition system an approximation of any language expressible in context-free rules. We chose to implement the constraints in the recognition system in the form of a finite automaton (FA) similar to those described in [4] and [1].

At the first stage in generating a grammar, we use a context-free notation augmented with variables in order to simplify the process of describing a language For example, this notation would allow a rule that says a noun phrase of any number can be replaced by an article and a noun of the same number, whereas ordinary context-free notation would require two rules that are identical except that one would be for singular number and the other for plural.

Our system first translates the augmented notation into ordinary context-free rules and then constructs a FA based on these rules. While it is true that context-free grammars can accept recursive languages which finite automata cannot, finite automata can approximate recursion by setting upper limits on the number of levels of recursion allowed. Such an approximation is reasonable for most task languages, since spoken sentences do not ordinarily use more than a few levels of recursion.

In our recognition system, the automaton is used as follows. Associated with each transition in the FA is a hidden-Markov word model that is used to compute the probability of a spectral sequence given the occurrence of the word at that place in the grammar The recognition algorithm with this grammar is only slightly different from the version of the algorithm that allows any sequence of words [2]. For each 10 ms frame of the input speech, the scores for all the word models in the FA network are updated according to a modified Baum-Welch algorithm. The score for the start state of the FA is unity and the score for every other FA state is simply the maximum of all the word model scores that enter the state along FA transitions This state score in turn, is propagated to the beginning of all the word

models on transitions leaving the state, to be used as the new initial score for those models. In this way the recognizer only considers grammatical sequences of words. Maintained throughout this scoring process are traceback pointers that indicate for each state and each time the word model that produced the best score to enter the state. Once an utterance is thus processed, it is a simple matter to follow these pointers back through the network to find the highest scoring sequence of words.

One potential difficulty with a FA grammar for recognition stems from the fact that, ordinarily, computation is proportional to the number of transitions in the FA. This number can become quite large for complex languages. However, in our experience with grammars for the Email task, a simple time-synchronous search with pruning [5] effectively reduces the computation to less than that for the algorithm that does not use a grammar, without affecting performance.

### 5.2 Description of the Grammars and Methodology

We compare here the effects on performance of grammars differing in which set of sentences they are intended to cover (the full set of test sentences or the commands only) and along a dimension we call tight-loose, which refers to an estimate of how much over-generation is produced by the grammar. "Tight" grammars have very little over-generation (generation of sentences that are considered ungrammatical) and, because of these tighter constraints, tend to have fewer choices at various points in the grammar, i.e., smaller perplexity. "Loose" grammars, on the other hand, have a great deal of over-generation and greater perplexity (larger sets of choices at various states). The loose grammars developed here are loose in that, for example, no number, tense, case or semantic agreement is required.

The grammars we have investigated so far include a tight and a loose grammar for commands (COM-T and COM-L, respectively) and a loose grammar that covers both commands and questions (SENT-L). In addition, we have used another grammar that is tighter than SENT-L (and hence is called SENT-T), but only in aspects that would otherwise put into similar grammatical distribution large sets of minimal pairs. For example, singular versus plural nouns, the cardinals versus ordinals, or verb tenses all involve large sets of acoustically similar items. This fact can pose a problem for recognition if the grammar allows many sequences in which one member of the pair can be substituted for the other. On the other hand, distinguishing verbs on the basis of which objects they take reduces perplexity without necessarily reducing the number of acoustically similar competing words.

Table I shows the relevant attributes of the grammars investigated. For comparison, the results for no grammar (the trivial grammar that allows any lexical item to occur anywhere) are also included. The table includes the number of arcs (a rough measure of size, and is related to computation time), the three estimates of perplexity (Maximum Perplexity, Test Set Branching Factor, and Uniform Branching). This table also shows the number of words and number of sentences on which each grammar was tested, and the performance for each. Word accuracy here is computed as the sum of all errors (insertions + deletions + substitutions) divided by the sum (total words + insertions). Sentence accuracy is also included in order to show that a few percentage points difference in word accuracy can result in much larger differences in the number of correctly recognized sentences, a number that is no doubt very important to potential users.

Since we had used 30 of the 100 test sentences in previous experiments and modified our system as a function of those results, we used only the subset of 70 remaining sentences for the performance figures reported here. In order to compare the tight and loose versions of the grammars, performance was assessed using the intersection of the sentences parsed by each grammar. Results are based on using the phone-left-and-right word-model discussed in [2].

### 5.3 Results and Discussion

Figures 2a (commands only) and 2b (commands and questions) show graphically the word accuracy figures of Table I associated with each grammar. Performance is plotted as a function of the perplexity estimates used. As can be seen, these grammars differ in their effects on performance. Further, when two grammars that cover the same set of sentences are compared (COM-T versus COM-L or SENT-T versus SENT-L), the more constrained grammar has significantly better word accuracy than the less constrained one. Tightening of the command grammar improved performance from 95.5% to 98.4%. Tightening of the sentence grammar improved performance from 96.2% to 98.2%. Word accuracy, again, includes as errors all insertions, deletions and substitutions. Further, it appears that grammatical constraints that take into account acoustic similarity

TABLE I

Properties of the Grammars

| GRAMMAR | COM-L | COM-T | SENT-L | SENT-T | NONE |
|---|---|---|---|---|---|
| Number of arcs | 836 | 7167 | 2547 | 3771 | |
| Maximum Perplexity | 58 | 19 | 75 | 60 | 334 |
| Test Set Branching | 40 | 18 | 47 | 31 | 334 |
| Uniform Branching | 19 | 9 | 22 | 19 | 334 |
| Words in test set | 183 | 183 | 438 | 438 | 492 |
| Sentences in test set | 27 | 27 | 63 | 63 | 70 |
| Sentence accuracy | 72.9% | 90.1% | 80.5% | 90.2% | 36.7% |
| Word Accuracy | 95.5% | 98.4% | 96.2% | 98.2% | 86.6% |

Comparison of the various grammars used for the commands (tight coverage, COM-T; loose coverage, COM-L) and the commands plus questions (tight coverage, SENT-T; loose coverage, SENT-L). Word accuracy here is computed as (insertions + deletions + substitutions) divided by (total words + insertions).

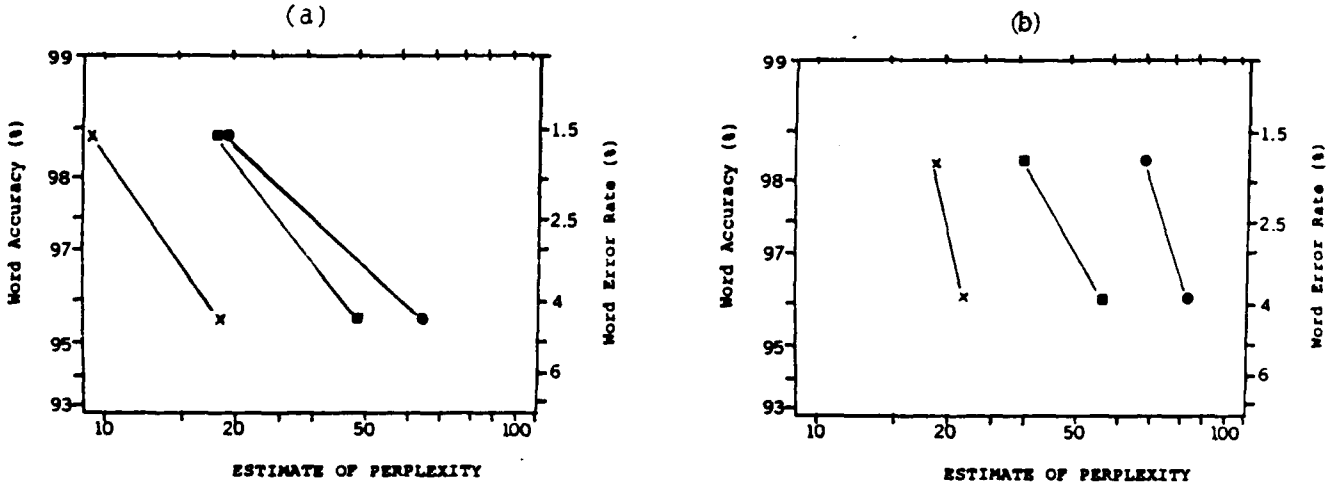



Figure 2: Performance with Grammars. Plotted is performance, (insertions + deletions + substitutions) divided by (number of words + insertions), as a function of perplexity as estimated by the uniform branching assumption (X), the test set branching factor (squares), and the maximum perplexity (circles). (a) The tightly constrained command grammar (COM-T) and its loose counterpart (COM-L). (b) The tightly constrained sentence grammar (SENT-T) and its loose counterpart (SENT-L), which considers acoustic similarity.

improve performance more than those that do not for comparable estimated perplexity the SENT-L grammar improves performance more than its estimated perplexity would predict if acoustic similarity had not been an important factor

An analysis of the recognition errors using these various grammars reveals that, in general, acoustically similar items are confused It does not appear that function words are more often involved in the errors than content words A large percentage of our errors (32% for SENT-T) involve "the" and "a", which happen to be function words However, no other function words show this pattern We believe that "the" and "a" show up more often in the errors NOT because they are function words, but because they are (1) acoustically similar, (2) have similar grammatical distributions, and (3) are very frequent words in these sentences. Assuming that we cannot change their acoustic similarity or their lexical frequency, improving performance on these words requires a more constrained specification of their distribution in the linguistic model. It is possible that semantic, pragmatic or discourse models could separate the two distributions, given a well-defined task environment.

# 6 CONCLUSIONS AND FUTURE RESEARCH

We have implemented and tested methods of combining grammatical and acoustic knowledge sources in our recognition algorithm. We find that the use of grammatical constraints can decrease the error rate by a factor of more than six. This result corresponds to a word accuracy (counting all insertions, substitutions and deletions as errors) of more than 98% for the Email task. Reducing the number of words considered by the recognizer boosts performance, even when the amount of training per word is fixed. We have presented various estimates of grammatical perplexity and shown that performance improves as estimated perplexity decreases for a given task. Our experience with a grammar that focuses only on syntactic constraints in acoustically confusable portions of the grammar demonstrates the importance of acoustic similarity in predicting performance accurately and in improving recognition performance

# References


1 Y.L. Chow, R.M. Schwartz, S. Roucos, O.A. Kimball, P.J. Price, G.F. Kubala, M.O. Dunham, M.A. Krasner, and J. Makhoul, "The Role of Word-Dependent Coarticulatory Effects in a Phoneme-Based Speech Recognition System", *IEEE Int Conf Acoust Speech, Signal Processing*, Tokyo, Japan April 1986.

2. L.R. Bahl, F. Jelinek, and R.L. Mercer, "A Maximum Likelihood Approach to Continuous Speech Recognition", *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol. PAMI-5, No 2, March 1983, pp. 179-190.

3 M.M. Sonhdi and S.E. Levinson, "Computing Relative Redundancy to Measure Grammatical Constraint in Speech Recognition Tasks", *IEEE Int. Conf. Acoust., Speech, Signal Processing*, Tulsa, OK, April 1978, pp. 409-412.

4 R.G. Goodman, *Analysis of Languages for Man-Machine Communication*, PhD dissertation, Carnegie-Mellon Univeristy, May 1976.

5. S.E. Levinson, "Structural Methods in Automatic Speech Recognition", *Proc. IEEE*, Vol. 73, No 11 November 1985, pp. 1625-1650. Special Issue on Man-Machine Communications

6. B.T. Lowerre, *The Harpy Speech Recognition System*, PhD dissertation, Carnegie-Mellon University, 1976.

IEEE International Conference on Acoustics, Speech, and Signal Processing, April 6-9, 1987, Dallas, Texas, pp. 89-93.

# BYBLOS: The BBN Continuous Speech Recognition System

Y.L. Chow, M.O. Dunham, O.A. Kimball, M.A. Krasner,
G.F. Kubala, J. Makhoul, P.J. Price, S. Roucos,
and R.M. Schwartz

BBN Laboratories Incorporated
10 Moulton Street
Cambridge, MA 02239

## Abstract

In this paper, we describe BYBLOS, the BBN continuous speech recognition system. The system, designed for large vocabulary applications, integrates acoustic, phonetic, lexical, and linguistic knowledge sources to achieve high recognition performance. The basic approach, as described in previous papers [1, 2], makes extensive use of robust context-dependent models of phonetic coarticulation using Hidden Markov Models (HMM). We describe the components of the BYBLOS system, including: signal processing frontend, dictionary, phonetic model training system, word model generator, grammar and decoder. In recognition experiments, we demonstrate consistently high word recognition performance on continuous speech across: speakers, task domains, and grammars of varying complexity. In speaker-dependent mode, where 15 minutes of speech is required for training to a speaker, 98.5% word accuracy has been achieved in continuous speech for a 350-word task, using grammars with perplexity ranging from 30 to 60. With only 15 seconds of training speech we demonstrate performance of 97% using a grammar.

## 1. Introduction

Speech is a natural and convenient form of communication between man and machine. The speech signal, however, is inherently variable and highly encoded. Vast differences occur in the realizations of speech units related to context, style of speech, dialect, talker. This makes the task of large vocabulary continuous speech recognition (CSR) by machine a very difficult one. Fortunately, speech is also structured and redundant: information about the linguistic content in the speech signal is often present at the various linguistic levels. To achieve acceptable performance, the recognition system must be able to exploit the redundancy inherent in the speech signal by bringing multiple sources of knowledge to bear. In general, these can include: acoustic-phonetic, phonological, lexical, syntactic, semantic and pragmatic knowledge sources (KS). In addition to designing representations for these KSs, methodologies must be developed for interfacing them and combining them into a uniform structure. An effective and coherent search strategy can then be applied based on global decision criteria. Practical issues that need to be resolved include computation and memory requirements, and how they could be traded off to obtain the desired combination of speed and performance.

In BYBLOS, we have explored many issues that arise in designing a large and complex system for continuous speech recognition. This paper is organized as follows. Section 2 gives an overview of the BYBLOS system. Section 3 describes our signal processing frontend. Section 4 describes the trainer system used for phonetic model knowledge acquisition. Section 5 describes the word model generator module that compiles word HMMs for each lexical item. Section 6 describes the syntactic/grammatical knowledge source that operates on a set of context-free rules describing the task domain to produce an equivalent finite state automaton used in the recognizer. Section 7 describes the BYBLOS recognition decoder using combined multiple sources of knowledge. Finally, Section 8 presents some figures and discussions on BYBLOS recognition performance.

## 2. Byblos System Overview

Figure 1 is a block diagram of the BYBLOS continuous speech recognition system. We show the different modules and knowledge sources that comprise the complete system, the arrows indicating the flow of module/KS interactions. The modules are represented by rectangular boxes. They are, starting from the top: Trainer, Word Model Generator, and Decoder. Also shown are the knowledge sources, which are represented by the ellipses. They include: Acoustic-Phonetic, Lexical, and Grammatic knowledge sources. We will describe briefly the various modules and how they interact with the various KSs.

Acoustic-Phonetic KS

The Trainer module is used for the acquisition of the acoustic-phonetic knowledge source. It takes as input a dictionary and training speech and text, and produces a database of context-dependent HMMs of phonemes.

Lexical KS

The Word Model Generator module takes as input the phonetic models database, and compiles word models phonetic models. It uses the dictionary - the lexical KS, in which phonological rules of English are used to represent each lexical item in terms of their most likely phonetic spellings. The lexical KS imposes phonotactic contraints by allowing only legal sequences of phonemes to be hypothesized in the recognizer, reducing the search space and improves performance. The output of the Word Model Generator is a database of word models used in the recognizer.

Grammatical KS

More recently, we have been working on representation and integration of higher levels of knowledge sources into BYBLOS, including both syntactic and semantic KSs. By incorporating both of these KSs into BYBLOS in the form of a grammar into our recognizer, we demonstrate improved recognition performance. In Section 6, we describe the Grammatical KS in more detail.

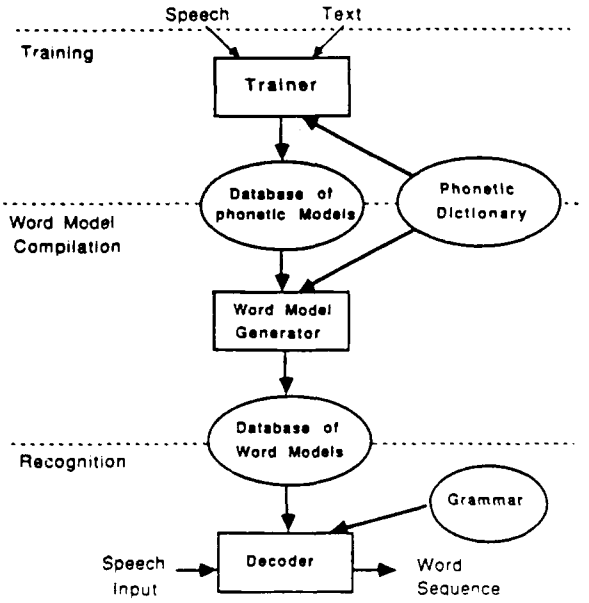



Figure 1: BYBLOS System Diagram.

## 3. Signal Processing and Analysis Component

The BYBLOS signal processing frontend performs feature extraction for the acoustic models used in recognition. Sentences are read directly into a close talking microphone in a natural but deliberate style in a normal office environment. The input speech is lowpass filtered at 10 kHz and sampled at 20 kHz. Fourteen Mel-frequency cepstral coefficients (MFCC) are computed from short-term spectra every 10 ms using a 20 ms analysis window. This MFCC feature vector is then vector quantized to an 8-bit (256 bins) representation. The vector quantization (VQ) codebook is computed using the k-means clustering algorithm with about 5 minutes of speech. We perform a variable-frame-rate (VFR) compression in which strings of up to 3 identical vector codes are compressed to a single observation code. We found this VFR procedure speeds up computation with no loss in performance.

## 4. Training/Acquisition Of Phonetic Coarticulation Models

The training system in BYBLOS acquires and estimates the phonetic coarticulation models used in recognition. Given that we model speech parameters as probabilistic functions of a hidden Markov chain, we make use of the Baum-Welch (also known as the Forward-Backward) algorithm [3] to estimate the parameters of the HMMs automatically from spoken speech and corresponding text transcription. For each training utterance, the training system takes speech and text, and builds a network of phonemes using the dictionary. It first builds the phonetic network for the word by using the phonetic transcription provided by the dictionary. The phonetic network is expanded into a triphone network so that each arc completely defines a phonetic context up to the triphone. These triphone networks of the word are then concatenated to form a single network for the sentence, which in general can take into account within word as well as across-word phonological effects. The training system then compiles a set of phonetic context models for each triphone arc in the network. It then runs the forward-backward algorithm to estimate the parameters of the phonetic context models. The Trainer operates in two modes: speaker-dependent and speaker-adapted. Associated with these two modes are two distinct methods for training the parameters of the hidden Markov models described below.

Speaker-Dependent

This is the algorithm used to find the parameters of the HMMs that maximizes the probability of the observed data given the model. This method produces HMMs that are finely tuned to a particular speaker, therefore in general would work well only for this speaker. Typically about 15 minutes of speech from a speaker is required for speaker-dependent training.

Speaker-Adapted

This is a new method of training that transforms HMM models of one speaker to model the speech of a second speaker [4]. This procedure estimates a probabilistic spectral mapping from a well-trained prototype speaker to a new speaker. Using this method it is possible for a new speaker to used the system with as little as 15 seconds of speech.

## 5. Word Model Generator

Prior to recognition, word HMMs are computed for each word in the vocabulary. The word model generator takes as input two objects: a database of phonetic HMMs as obtained in training, and a dictionary that contains phonetic spellings for each word. For each phoneme in each word of the lexicon, it first finds in the phonetic HMM database all the context models that are relevant to this phoneme in its particular phonetic environment. It then combines this set of phonetic models with appropriate weights to produce a single HMM for each phoneme in the word. This combination process saves computation by precompiling the many levels of phonetic context models that can occur for a given phonetic context into a single representation. The output of the word model generator is a database of word HMMs serving as the input to the decoder.

## 6. Grammatical Knowledge Source

To solve the CSR problem requires major advances in two areas: acoustic modeling and language modeling. A good acoustic model is essential in making fine phonetic distinctions when needed. However, it is not sufficient by itself to solve the CSR problem. In a complex task with large vocabulary where the number of hypothesized word candidates is large, the probability for acoustic confusability can be high, and the recognizer could make errors. A conceptually simple yet effective way to restrict the number of words that are allowed to be hypothesized, and therefore decrease probability of acoustic similarity, is to incorporate a grammar into the recognizer. It is well known that recognition performance improves as vocabulary size decreases. Similarly, when syntactical information is used to reduce the number of words that can legally follow a given sequence of words, a recognizer is expected to make fewer errors. The purpose for using a grammar then, is to improve recognition performance, with an added benefit of reducedcomputation.

Grammar Design and Implementation

We approach the implementation of a grammar in BYBLOS in two stages. First, we create a description of the task domain language using a modified context-free notation. Typically this description is based on a representative set of sentences that characterizes the task domain, and is designed to capture generalizations of the linguistic phenomena found in them. Second, we use a tool that transforms this description into structures in our recognizer that provide the corresponding grammatical constraints. This tool provides us with a general facility for capturing in BYBLOS an approximation of any language expressible in context-free grammars (CFG) expressed as context-free rules. We elected to implement the grammatical constraints in the form of a finite state automaton (FA) similar to those described in [5].

At the first stage in generating a grammar, we use a context-free notation augumented with variables in order to simplify the process of describing a language. For example, this notation would allow a rule that says a noun phrase of any number can be replaced by an article and a noun of the same number; ordinary context-free notation would require two rules that are identical except that one would be for singular number and the other for plural.

Our system first translates the augmented notation into ordinary CFGs and then constructs a FA based on these rules. Because context-free grammars can accept recursive languages and a FA cannot, recursion is approximated in the FA by limiting the number of levels of recursion. Such an approximation is reasonable for most task languages, since spoken sentences do not ordinarily use more than a few levels of recursion.

## 7. Recognition Search Strategy

Once the FA is compiled from the context-free description of the task domain, it is ready to be used in the decoder. An important characteristic of a recognizer is the search strategy that is used to find the word sequence that best matches the input speech. We believe that an optimum search strategy avoids making local decisions; the search decision should be made globally, based on scores from all the KSs. One such search paradigm is the one used in BYBLOS: the search is made top down, linguistically driven, with tightly coupled KSs.

The FA is convenient for deploying such a search strategy. It is used as follows in our recognizer. We associate with each transition in the FA a hidden Markov model for the word. This model is used to compute the probability of the acoustic event (sequence of VQ spectra) given the occurrence of the word at that place in the grammar. Before the start of recognition, the initial state of the FA where a legal sequence of words can begin is initialized to unity, and all the other states are initialized to zero. For each 10 ms frame of the input speech, the scores for the states in all the words in the FA network are updated using modified Baum-Welch algorithm [2]. In addition to state updates within a word, a word can have a score propagated to its initial state from its best scoring predecessor word. This simple state update operation is repeated every 10 ms for each FA transition until the end of the utterance is reached. The decoder output is then computed by tracing back through the FA network to find the highest scoring sequence of words that end in the terminal state of the FA.

One potential problem associated with using a FA grammar for recognition is that computation is expected to be proportional to the number of transitions in the FA. This number can be quite large for complex languages. However, in our experience with different grammars in our recognizer, we find that a beam search effectively reduces the computation to a very manageable level while maintaining the same performance as that of an exhaustive search.

## 8. Byblos Recognition Performance

In [4], we presented word recognition results for a 334-word electronic mail task. In speaker-dependent mode, we demonstrated performance of 90% across several speakers without the use of a grammar (i.e., branching factor of 334). Since then, we have tested the system along many dimensions: two task domains, FA grammars with varying perplexities, varying amounts of adaptation speech, and different speaker types. The results are tabulated in Figure 2. Below we describe the different conditions in more detail.

Task Domains

The two task domains tested are: Electronic Mail (EMAIL) and Naval Database Retrieval (FCCBMP). Both tasks have vocabulary sizes of approximately 350 word (334 for EMAIL, 354 for FCCBMP). A description of the task domain language was created using CFG. The CFGs were designed to capture generalizations of linguistic phenomena found in example task domain sentences.

Grammars

Two finite state grammars were generated for each task domain: Command and Sentence. The Command Grammar in each case was designed to cover only the command subset of

| Grammar/ Perplexity / Training Time | EMAIL | | FCCBMP | |
|---|---|---|---|---|
| | Command (20) | Sentence (30) | Command (22) | Sentence (30) |
| 15 minute | 98.4 | 98.8 | 99.6 | 99.5 |
| 2 minute | 97.9 | 94.9 | 96.6 | 96.2 |

**Figure 2:** BYBLOS Recognition Results. Two task domains (EMAIL and FCCBMP), two grammars for each task (Command and Sentence), and varying amounts of training speech (2 minutes and 15 minutes). Also shown are maximum perplexity measures for the grammars.

the language; the Sentence Grammar was designed to cover all of the language, which included both command and question type constructs. The maximum perplexity measures of the grammars, as proposed in [6], are shown in Figure 2. In both tasks, the sentence grammars have a higher perplexity than their command counterparts

Adaptation Time

As described in Section 2, The BYBLOS operate in two modes, speaker-dependent and speaker-adapted. In speaker-dependent mode, 15 minutes of training speech is required for a speaker. This mode in general will give word accuracy in the 98.5+ range. In the speaker-adaptive mode, anywhere from 2 minutes down to 15 seconds of speech from a new speaker is needed to "adapt" the HMM parameters to the new speaker. The performance in this case is 97%.

Speaker Type

We have tested BYBLOS on several speakers with different dialects, including a female speaker, a non-native speaker, and 3 naive (uncoached) speakers. The recognition results for these speakers showed little deviation typical male speakers of standard American dialects.

## 9. Summary

We have presented BYBLOS, a system for large vocabulary continuous speech recognition. We showed how we integrate multiple sources of knowledge to achieve high recognition performance. In recognition experiments, we demonstrated consistent performances across task domains, grammars, adaptation time, and speaker type.

We are currently working to improve various aspects of the system, including: a real time implementation of the recognizer, search strategy, acoustic modeling, and language modeling. In the future, we plan to work on integration of speech and natural language for speech understanding applications.

## Acknowledgement

This work was supported by the Defense Advanced Research Projects Agency and was monitored by the Space and Naval Warfare Systems Command under Contract No. N00039-85-C-0423.

## References

1. R.M. Schwartz, Y.L. Chow, O.A. Kimball, S. Roucos, M. Krasner, and J. Makhoul, "Context-Dependent Modeling for Acoustic-Phonetic Recognition of Continuous Speech", *IEEE Int. Conf. Acoust., Speech, Signal Processing*, Tampa, FL, March 1985, pp. 1205-1208, Paper No. 31.3.

2. Y.L. Chow, R.M. Schwartz, S. Roucos, O.A. Kimball, P.J. Price, G.F. Kubala, M.O. Dunham, M.A. Krasner, and J. Makhoul, "The Role of Word-Dependent Coarticulatory Effects in a Phoneme-Based Speech Recognition System", *IEEE Int. Conf. Acoust., Speech, Signal Processing*, Tokyo, Japan, April 1986, pp. 1593-1596, Paper No. 30.9.1.

3. L.R. Bahl, F. Jelinek, and R.L. Mercer, "A Maximum Likelihood Approach to Continuous Speech Recognition", *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol. PAMI-5, No. 2, March 1983, pp. 179-190.

4. R.M. Schwartz, Y.L. Chow, G.F. Kubala, "Rapid Speaker Adaptation using a Probabilistic Spectral Mapping", *IEEE Int. Conf. Acoust., Speech, Signal Processing*, Dallas, TX, April 1987, Elsewhere in these proceedings

5. R.G. Goodman, *Analysis of Languages for Man-Machine Communication*, PhD dissertation, Carnegie-Mellon University, May 1976.

6. M.M. Sondhi and S.E. Levinson, "Computing Relative Redundancy to Measure Grammatical Constraint in Speech Recognition Tasks", *IEEE Int. Conf. Acoust., Speech, Signal Processing*, Tulsa, OK, April 1978, pp. 409-412.

Presented at the IEEE International Conference on Acoustics, Speech, and Signal Processing, April 11-14, 1988, New York, N.Y. pp. 651-654

# The DARPA 1000-Word Resource Management Database for Continuous Speech Recognition

**Patti Price**
*BBN Laboratories, Inc*
*Cambridge, MA 02238*

**William M. Fisher**
*Texas Instruments, Inc*
*Dallas TX 75266*

**Jared Bernstein**
*SRI International*
*Menlo Park, CA 94025*

**David S. Pallett**
*National Bureau of Standards,*
*Gaithersburg, MD 20899*

## ABSTRACT

A database of continuous read speech has been designed and recorded within the DARPA Strategic Computing Speech Recognition Program. The data is intended for use in designing and evaluating algorithms for speaker-independent, speaker-adaptive speech, and speaker-dependent speech recognition. The data consists of read sentences appropriate to a naval resource management task built around existing interactive database and graphics programs. The 1000-word task vocabulary is intended to be logically complete and habitable. The database, which represents over 21,000 recorded utterances from 160 talkers with a variety of dialects, includes a partition of sentences and talkers for training and for testing purposes.

## 1 Introduction

The development of robust, reliable speech recognition systems depends on the availability of realistic, well-designed databases; the technical and commercial community can benefit greatly when different systems are evaluated with reference to the same benchmark material. The DARPA 1000-word resource management database was designed to provide such benchmark materials: it consists of consistent but unconfounded training and test materials that sample a realistic and habitable task domain, and cover a broad range of speakers. The goal of this database collection effort was to yield a set of data to promote the development of useful large-vocabulary, continuous speech recognition algorithms. We hope that this description will serve both to publicize the existence of the database and its availability for use in benchmark tests, and to describe the methods used in its construction.

The database includes materials appropriate to a naval resource management task. The 1000 vocabulary items and 2800 resource management sentences are based on interviews with naval personnel familiar with an existing test-bed database and accompanying software to access and display information. 160 subjects, representing a wide variety of US dialects, read sentence materials including 2 "dialect sentences" (i.e., sentences that contained many known dialect markers), 10 "rapid adaptation sentences" (designed to cover a variety of phonetic contexts), 2800 "resource management" sentences and 600 "spell-mode" phrases (words spoken and then spelled). The database is divided into a speaker-independent part and a speaker dependent part; both are divided into training and test portions. The test portions are further divided into equal sub parts for initial testing during system development ("development test"), and later evaluation ("evaluation test").

The methods build on and extend work by Leonard [3], Fisher *et al.* [2] and Bernstein, Kahn and Poza [1]. Original contributions of the current work include methods for designing the vocabulary and sentence set, speaker selection, and distribution of sentence material among the speakers.

The database design and implementation included: specification of a realistic and reasonable task domain, selection of a habitable 1000-word vocabulary, construction of sentences to represent the syntax, semantics, and phonology of the task, selection of a dialectally diverse set of subjects, assignment of subjects to sentences, recording of the subjects reading the sentences, and implementation of a system for the distribution and use of the database. These tasks are described in more detail below.

## 2 Task Design

### 2.1 Task Domain Specification

We chose a database query task because it is a natural place to use speech recognition technology as a human-machine interface. To define realistic constraints, and allow for eventual demonstrations of this technology, we based the task on the use of an existing, unclassified test-bed database and an interactive graphics program. The chosen task has the additional advantage that it has been the basis of much research and development in the natural language understanding community. The value of speech recognition technology is enhanced by its integration with a natural language understanding component.

The current phase of the DARPA speech recognition program specifies a 1000-word vocabulary. The test-bed database, however, has a substantially larger vocabulary size, and therefore had to be restricted. Our philosophy in selecting a 1000 word subset was to limit the number of database fields, rather than to limit the ways a user might access the information. The fields selected include information about various types of ships and associated properties: locations, propulsion types, fuel, sizes, fleet identifications, schedules, speeds, equipment availability and status. The interactive graphics commands include various ways of displaying maps and ship locations.

An initial set of 1200 resource management sentences came from: (1) preliminary interviews with naval personnel familiar with the test-bed database and the software for accessing it, and (2) systematic coverage of the database fields, subject to review by the naval personnel in follow-up interviews. These sentences were intended to provide wide coverage of the syntactic and semantic attributes of expected sentences, rather than expected relative frequencies of such sentences. Sentences were not filtered on the basis of "grammaticality", and therefore include, for example, instances of *the* deletion, lack of number agreement

between subject and verb, and many cases of ellipsis (i.e, omission of words required for strict grammaticality but not for comprehension, as in the deletion of the second instance of *speed* in *Is the Clerk's speed greater than the Ajax's speed*.

## 2.2 Vocabulary

The vocabulary was determined by collecting all words in the 1200 initial resource management sentences. If eventual users are expected to stay within the defined vocabulary, it should be, in some sense, grammatically, logically and semantically complete. Therefore, words were added so that the vocabulary included: (1) both singular and plural forms of nouns, (2) words required for all cardinal numbers less than a million, (3) words required for all ordinals needed for dates, (4) infinitive, present and past participle verb forms, (5) all months and days of the week. In addition, items were added for semantic "completeness". For example, since *high* occurred, *low*, *higher*, *highest*, *lower*, and *lowest* were added. The vocabulary was then completed by adding enough open class items to cover 33 ports, 26 other land locations, 26 bodies of water, and 100 ship names (in both nominative and possessive forms).

Since these sentences were to be read by naive subjects not familiar with the task domain or the database, the vocabulary was revised: some open class items were replaced with others thought to be easier to pronounce (*Sea of Japan* for *Sea of Okhotsk*), and spellings of some technical terms were changed to clarify the pronunciation (*TASSEM* for the acronym *TASM*).

## 2.3 Sentence Materials

The 1200 initial resource management sentences had some disadvantages: they included many slight variations of the same sentence (e.g., only a ship name changed or *the* deleted), and the vocabulary items were not evenly represented (the naval personnel interviewed tended to use only one or two ship names, for example, in all their examples). Further, we felt that far more than 1200 sentences would be needed to represent the vocabulary items and phonetic contexts of the task. Therefore, the initial 1200 sentences were reduced to a set of 950 unique surface semantic-syntactic patterns that were then used to generate 2800 sentences with excellent coverage of the vocabulary items.

The replacements included the replacement of instances of specific ship names with the variable [*shipname*], and of many instances of *the* with the variable [*optthe*] (to indicate optional *the*). About 300 such variables (indicated here by square brackets to distinguish them from vocabulary items) were defined and and used to replace specific instances.

In the two following examples, included to give an idea of the degree of abstraction involved, the variable definitions are: [*what-is*] ⇒ *what is*, *what's*; [*shipname's*] ⇒ *Kirk's*, *Fox's*, etc.; [*gross-average*] ⇒ *gross*, *average*; [*long-metric*] ⇒ *long*, *metric*; [*show-list*] ⇒ *show*, *list*, *show me*, etc.; [*ships*] ⇒ *carriers*, *cruisers*, etc.; [*water-place*] ⇒ *Indian Ocean*, *Sea of Japan*, etc.; [*date*] ⇒ *March 4th*, *2 June 1987*, etc.

1. [*what-is*] [*optthe*] [*shipname's*] [*gross-average*] *displacement*
2. [*show-list*] *optthe ships in water-place date*

After replacement of instances with variables in the 1200 sentences, duplicates were removed, yielding 950 sentence patterns. The patterns were ordered such that those with the most unique words or classes appeared first in the list.

The 950 sentence patterns generated 2800 sentences in three passes of substitution of an instance for each variable. A counter associated with each variable determined which instance should be used for each substitution. The patterns thus generated a set of sentences that systematically covered the vocabulary items. After removal of duplicates, there were 2835 sentences. The 35 longest sentences were removed; the remaining 2800 were hand edited to remove infelicities that could arise from the procedure (such as *one carriers* generated from [*cardinal*] [*ships*]). The first 600 sentences generated were designated **training** sentences; the ordering of the patterns and the generation procedure resulted in good coverage of the vocabulary: these 600 sentences cover 97% of the vocabulary items.

In between the concept of speaker-independence (requiring no new data from new speakers) and speaker-dependence (requiring a great deal of data from each new speaker) is the concept of speaker-adaptation (requiring a small amount of data from each new speaker). For use in speaker-adaptation technologies we have provided 10 "rapid adaptation" sentences, designed to provide a broad and representative sample of the speaker's production of phonemes and phoneme sequences of the 2800 resource management sentences. The goal was to provide embedded sets of one, two, five and ten sentences that each had the best coverage (for its size) of the relevant phonemic material. Thus, the first is the best adaptation sentence, the second sentence, when added to the first, is the best combination of two sentences according to the same coverage criteria, and so on up to ten.

A coverage score was calculated for each phoneme and phoneme pair in a sentence based on the observed frequency of the phoneme or phoneme pair in the 2800 sentences, but breadth of coverage was promoted by dividing the observed frequency of each phoneme or phoneme pair by a factor (we used 3.0) each time it was used in the material currently having a score calculated. In order to inhibit the tendency for the longest (and most difficult to read) sentences from being selected, we normalized by dividing the score by sentence length. The resulting adaptation sentences are listed in the appendix.

For the "spell-mode" utterances, 600 words were selected from the 1000 vocabulary items; the 400 words not selected were inflected variants of those chosen.

# 3 Subject Selection and Recording

## 3.1 Subject Selection

On the basis of demographic and phonetic characteristics, 160 subjects were selected from a set of 630 adults who had participated in an earlier database effort [2]. These 630 native speakers of English (70% male, 30% female) with no apparent speech problems formed a relatively balanced geographic sample of the United States. As a group, the subjects were young, well educated, and White: 63% in their twenties, 78% with a bachelors degree and 1% Black. Each speaker was identified with one of eight geographic regions of origin: New England, New York, Northern, North Midland, South Midland, Southern, Western, or Army Brat (people who moved around a lot while growing up).

Among other material, each of these 630 subjects had recorded two dialect-shibboleth sentences (i.e., sentences containing several instances of words regarded as a criterion for distinguishing members of dialect groups). These sentences, included in the appendix, were hand-transcribed and used to derive a phonetic profile of each speaker as to phonology, voice quality, and manner of speaking. The 630 speakers were automatically divided into 20 clusters according to their pronunciation of several consonants, speaking rate, F0, and phonation quality. From these 630 speakers (now identified by phonetic cluster, geographic origin and demographic characteristics) 160 were selected for the speaker-independent part of the database, and 12 for the speaker-dependent part.

The 160 speaker-independent subjects were chosen to satisfy the following constraints, in order: 1) even distribution of subjects over four geographic regions (NE-NY, Midland, South, North-West-or-Army) and over the 20 clusters derived from observed phonetic characteristics; 2) 70% male, 30% female. These constraints are satisfied in the subject selection, and each major division of the database (training, development test and evaluation test) have similar distributions across sex and geographic origin.

The 12 speaker-dependent subjects were chosen to satisfy the following constraints: 1) representation of each of the 12 largest phonetic clusters; 2) seven male, five female; and 3) geographical representation as follows: one each from New York and New England, and two each from Northern, North Midland, South Midland, Southern, and Western. Of the 12 selected speakers, 11 were from the speaker-independent part of the database, and all were relatively fluent readers with no obvious speech problems.

## 3.2 Subject-Sentence Assignment

Both the speaker-independent and speaker-dependent parts of the database are divided into sets for training, development test and evaluation test.

In the speaker-independent training part of the database, 80 speakers each read 57 sentences (40 resource management sentences, the 2 dialect sentences, and 15 spell-mode phrases). 1600 distinct resource management sentences were covered in this part of the database; any given sentence was recorded by two subjects. The distribution of sentences to speakers was arbitrary, except that no sentence was read twice by the same subject. Each of the 80 speakers read 15 spell-mode phrases, yielding 1200 productions covering 300 unique words. Each spell mode phrase in this part was read by 4 speakers.

In the speaker-independent development test set and evaluation test set, 40 speakers each read 30 resource management sentences, the 2 dialect sentences, the 10 rapid adaptation sentences, and 15 spell-mode phrases. 600 resource management sentences were randomly selected for each test and assigned to the 1200 available productions (40 speakers times 30 sentences), yielding two productions per sentence, as in the training phase. Similarly, in each test set, 150 spell mode phrases were selected and assigned to the 600 available spell-mode productions.

The following table illustrates the structure of the speaker-independent part of the database. The numbers indicate how many sentences each subject read. The total number of resource management sentences covered by each subset of the database is indicated in parentheses. These are referred to as "types" in the table in distinction to sentence tokens, or productions by a particular speaker. In all, for the speaker-independent database, 9120 sentences were recorded (1560 for training, 2280 for development test, and 2280 for evaluation test). Note that, this being the speaker-**independent** database portion, the training subjects do not overlap with those in the test parts of the database.

**SPEAKER-INDEPENDENT DATABASE**

| | training | | development test | | evaluation test | |
|---|---|---|---|---|---|---|
| No. Subjects | 80 | | 40 | | 40 | |
| No. Sentences (types) | | | | | | |
| Resource Management | 40 | (1600) | 30 | (600) | 30 | (600) |
| Dialect | 2 | (2) | 2 | (2) | 2 | (2) |
| Adaptation | 0 | (0) | 10 | (10) | 10 | (10) |
| Spell-mode | 15 | (300) | 15 | (150) | 15 | (150) |
| TOTALS | 57 | (1902) | 57 | (762) | 57 | (762) |

For the speaker-dependent training portion of the database, each of 12 subjects read the 600 resource management training sentences, the 2 dialect sentences, the 10 rapid adaptation sentences, and a selection of 100 spell-mode phrases. The 1200 spell-mode readings covered 300 word types, with 4 productions per word.

In the speaker-dependent test portion of the database, these same 12 speakers each read 100 resource management sentences for the development-test part of the database and another 100 resource management sentences for the evaluation-test part, as well as 50 spell-mode phrases. From the 2200 resource management sentences not read in the training phase, two random selections of 600 sentences were made, one for the development test and one for the evaluation test portion. Distributing these over the productions available in each gives 2 utterances per sentence. Similarly, two random selections of 150 words each were made from the pool of 600 spell-mode phrases for the development and evaluation test sets. Distributing these over the 600 readings available yields 4 productions per word.

The following table illustrates the structure of the speaker-dependent part of the database. Again, the total number of different resource management sentences ("types") covered in each subset is indicated in parentheses after the number indicating how many sentences were read by each subject. In all, for the speaker-dependent database, 12,144 utterances were recorded (8544 for training, 1800 for development test, and 1800 for evaluation test). As is appropriate for a speaker-dependent database, the speakers in the training set are the same as the speakers in the test set.

**SPEAKER-DEPENDENT DATABASE**

| | training | | development test | | evaluation test | |
|---|---|---|---|---|---|---|
| No. Subjects | 12 | | 12 | | 12 | |
| No. Sentences (types) | | | | | | |
| Resource Management | 600 | (600) | 100 | (600) | 100 | (600) |
| Dialect | 2 | (2) | 0 | (0) | 0 | (0) |
| Adaptation | 10 | (10) | 0 | (0) | 0 | (0) |
| Spell-mode | 100 | (300) | 50 | (150) | 50 | (150) |
| TOTALS | 712 | (912) | 150 | (750) | 150 | (750) |

### 3.3 Recording Procedure

The utterances were digitally recorded in a sound-isolated recording booth on two tracks: one from a Sennheiser HMD414 headset noise-cancelling microphone, and the other from a B&K 4165 one-half inch pressure microphone positioned 30 cm from the subject's lips, off-center at a 20 degree angle. The material was digitized at 20,000 16-bit samples per second per channel, and then down-sampled to 16,000 kHz.

Prompts appeared in double-high letters on a screen for the subject to read. After the recording, both the subject and the director of the recording session listened to the utterances and re-recorded those with detected errors. Any pronunciation considered normal by the subject was accepted.

## 4 Database Availability and Use

This database, which is intended for use in designing and evaluating algorithms for speech recognition, is being made available to provide: (1) a carefully structured research resource, and (2) benchmarks for performance evaluation to judge both incremental progress and relative performance.

At present only the data from the Sennheiser microphone is available. This material alone amounts to approximately 930 Megabytes (MB) of data for the speaker-dependent subset and 640 MB for the speaker-independent subset, with an additional 460 MB included in the spell-mode subset. The down-sampled (16 kHz) data in Unix "tar" format (6250 bpi) can be made available on a loan, copy and return basis.

To provide benchmark test facilities, a set of procedures and a uniform scoring software package have been developed at the National Bureau of Standards (NBS). The scoring software implements a dynamic programming string alignment on the orthographic representations for the reference sentences and for the system outputs. Comparable scoring necessitated agreement on a standard orthographic representation for each vocabulary item. The scoring software and testing procedure are being used in the DARPA program for performance evaluation, and are available to the general public on request [4].

For those organizations wishing to determine and report performance data corresponding to that reported by DARPA program participants, NBS can provide test material used in DARPA benchmark tests [4]. If the results are to be publicly reported, it is required that the summary statistics be obtained using the NBS scoring software, and that copies of system output for these tests be made available to NBS.

## 5 Conclusion

For DARPA program participants, this database has proven useful in the design and evaluation of speaker independent, speaker-adaptive, and speaker-dependent speech recognition technologies; we hope it will be useful to others as well. Similarly, the methods developed for its design and collection should prove useful in the development of similar databases.

We have described the characteristics of the DARPA 1000-word resource management database: the task domain, the vocabulary, the sentence materials, the subjects, the division into training and testing portions. We have also described the steps involved in creating this database, including the recording procedure and new methods for designing the vocabulary and sentence set, speaker selection, and distribution of sentence materials among the speakers. In addition, we have outlined procedures for obtaining the database and for using it as a benchmark. Further details on each of these areas will be made available with the database.

**Acknowledgements.** This effort has been a collaborative effort that involved not just the authors, but the DARPA speech recognition community in general. However, chief responsibility for the various tasks required by the project was assigned as follows: BBN - task design, vocabulary selection and sentence construction; SRI - subject selection and dialect sentences; TI - subject-sentence assignment and recording of data; NBS - distribution and evaluation methods. We gratefully acknowledge the naval experts who helped us and the DARPA Strategic Computing Speech Recognition Program for funding this effort (contract numbers N00039-85-C-0123, N00039-85-C-0338 and N00039-85-C0302 monitored by SPAWAR, and, for NBS, DARPA order number 6079).

## References

[1] Bernstein, J., M. Kahn and T. Poza (1985) "Speaker sampling for enhanced diversity," *IEEE ICASSP-85*, paper 41.2.

[2] Fisher, W., V. Zue, J. Bernstein and D. Pallett (1987) "An acoustic-phonetic database," *J. Acoust. Soc. Am., Vol 81, Suppl. 1*, abstract OO1.

[3] Leonard, R. G. (1984) "A database for speaker-independent digit recognition," *IEEE ICASSP-84*, paper 42.11.

[4] For further information on availability of the database, test procedures and scoring software, contact D. S. Pallett, Room A216 Technology Building, National Bureau of Standards, Gaithersburg, MD, 20899. Telephone: (301) 975-2935.

### APPENDIX

#### Dialect-Shibboleth Sentences

1. She had your dark suit in greasy wash water all year.
2. Don't ask me to carry an oily rag like that.

#### Rapid Adaptation Sentences

1. Show locations and C-ratings for all deployed subs that were in their home ports April 5.
2. List the cruisers in Persian Sea that have casualty reports earlier than Jarrett's oldest one.
3. Display posits for the hooked track with chart switches set to their default values.
4. What is England's estimated time of arrival at Townsville?
5. How many ships were in Galveston May 3rd?
6. Draw a chart centered around Fox using stereographic projection.
7. How many long tons is the average displacement of ships in Bering Strait?
8. What vessel wasn't downgraded on training readiness during July?
9. Show the same display increasing letter size to the maximum value.
10. Is Puffer's remaining fuel sufficient to arrive in port at the present speed?

Presented at the IEEE International Conference on Acoustics, Speech, and Signal Processing, April 11-14, 1988, New York, N.Y. pp. 267-270

# STATISTICAL LANGUAGE MODELING USING A SMALL CORPUS FROM AN APPLICATION DOMAIN [1]

*Jan Robin Rohlicek* *Yen-Lu Chow* *Salim Roucos*

BBN Laboratories Incorporated
Cambridge MA 02238

## ABSTRACT

Statistical language models has been successfully used to improve performance of continuous speech recognition algorithms. Application of such techniques is difficult when only a small training corpus is available. This paper presents an approach for dealing with limited training available from the DARPA *resource management* domain. An initial training corpus of sentences was abstracted by replacing sentence fragments or *phrases* with variables. This training corpus of phrase sequences was used to derive parameters a Markov model. The probability of a word sequence is then decomposed into the probability of possible phrase sequences and the probabilities of the word sequences within each of the phrases.

Initial results obtained on 150 utterances from six speakers in the DARPA database indicate that this language modeling technique has potential for improved recognition performance. Furthermore, this approach provides a framework for incorporating linguistic knowledge into statistical language models.

## 1 INTRODUCTION

This paper addresses the use of statistical language modeling techniques in continuous speech recognition in the DARPA 1000-word *naval resource management* application domain [5]. This application involves the recognition of "natural" speech queries to an interactive database system. As will be discussed below, the "language" which will be used is unknown and a large training corpus is not available. Straightforward application of statistical language modeling techniques is therefore difficult. However, a language model is required to obtain very good recognition performance.

Language models provide a way of assigning likelihoods to word sequences in a language. The combination of such a measure with a measure of the acoustic likelihood of a word sequence has been shown to give good recognition performance in many applications. Several approaches have been successfully employed for languages of various complexity and various sizes of training corpus (for example [2]).

In certain restricted domains, finite state grammars have been used with considerable success (see [4] for example). In this case, the likelihood of a word sequence is a binary decision — a sequence is either parsed in the grammar or it is not in the allowable language. The extent to which the actual word sequences in the application are parsed by the grammar is termed *coverage*. When the language is known and not complex, the coverage is generally high and the constraints are well modeled by the grammar.

In the case of large vocabularies (> 1000 words) and "natural" language input one approach taken is the specification of formal grammars which describe the syntactic and semantic constraints of the domain [6]. The important issue is then the extent to which this grammar provides sufficient coverage while ruling out invalid word sequences. It has been found that it is difficult to achieve a high degree of coverage however. Recognition performance is generally high on sequences parsed by the grammar. However, when coverage of the valid word sequences is not high, then the language model actually introduces errors by not allowing valid word sequences.

To overcome the performance constraints imposed by poor coverage, statistical language models can be used. When a large training corpus is available, the parameters of a statistical language model can be determined. To the extent that the training corpus is representative of the real application, such techniques provide good performance [1]. Furthermore, since no binary decision as to the validity of a word sequence is necessary, the method is less "brittle" than the formal grammar techniques.

In the domain of interest in this paper, the language is not sufficiently well defined to allow the use of a finite-state grammar which both captures the constraints of the domain and is of reasonable size. Furthermore, there is no adequate training corpus for construction of a straightforward statistical model to characterize the word sequences. Due to the natural language interface, a grammar describing the complete language is very complex. Also, it is difficult to evaluate the extent to which any particular grammar

[1] This research was supported by the Defense Advanced Research Projects Agency under contract N00039-85-C-0423 monitored by SPAWAR

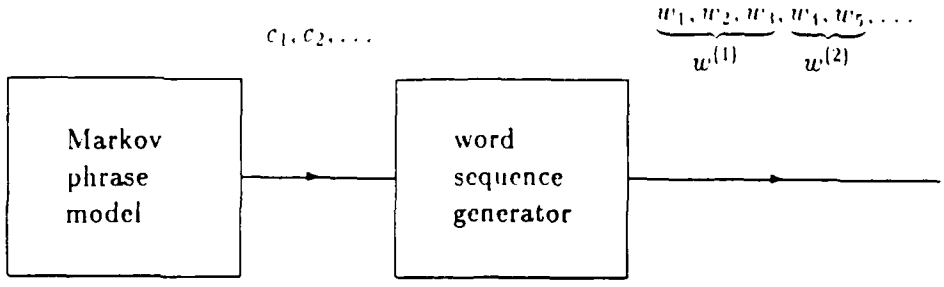



Figure 1: Word sequence model

covers sentences of the ultimate application domain. The complexity of the language suggests a statistical approach. However, since the application does not yet exist, a truly representative training corpus is not available. Furthermore, we feel that due do heavy use of jargon and unusual sentence structure, any attempt to use a training corpus from another domain, such as general English text, would be ineffective.

The approach described in this paper attempts to incorporate some linguistic knowledge of the structure of the language into a probabilistic framework. Using this approach, we will show very good performance can be obtained when the algorithm is evaluated on sentences which are independent of those used in construction of the statistical model.

In the next section, the basic structure of the model is described followed by a description of the training method employed. In Section 3, the results on six speakers from the DARPA database are presented. Finally, Section 4 contains a short discussion and concluding remarks.

## 2 APPROACH

### 2.1 Language Model Structure

The principle goal in the design of the probabilistic language model is to allow the estimation of robust model parameters from the modest training corpus which is available. A Markov model used to generate word sequences directly has too many parameters (the transition probabilities) to be estimated reliably from the limited training corpus. By considering a simpler model, which has fewer parameters associated with it, robust estimates might be obtainable. Furthermore, some linguistic structure can be identified, and this structure is incorporated into the model.

The model for the generation of a word sequence is composed of two part (Figure 1). First, a sequence of *phrase* variables $c_1, c_2, \ldots$, is generated as a Markov chain. Then, for each phrase $c_i$, a sequence of words $w^{(i)}$ is generated, independent of the phrases $c_j$, $j \neq i$. The probability of a phrase sequence $c_1, c_2, \ldots c_N$ is

$$\Pr(c_1, \ldots, c_N) = \Pr(c_1)\Pr(c_2|c_1)\cdots\Pr(c_N|c_1, \ldots, c_{N-1})$$

The probability of the phrase sequence and the word sequence $w_1, w_2, \ldots, w_n$ is then

$$\Pr(c_1, \ldots, c_N, w_1, \ldots, w_n) = \sum_{N, w^{(1)}, \ldots, w^{(N)}} \prod_{i=1}^{N} \Pr\left(w^{(i)} \mid c_i\right) \Pr(c_i \mid c_1, \ldots, c_{i-1})$$

where the sum is effectively over the possible segmentations of the word sequence into the phrases. *Note that since any $w^{(i)}$ might be a null expansion of a phrase, this representation of the probability in fact has an infinite number of terms.*

Using this structure, we identify phrases based on syntactic and semantic components of the language. For example, typical phrases include "open" set classes such as ship names or complex expressions such as dates. Also, to complete the coverage of the language, single word phrases are also allowed. Associated with each phrase is a small finite state grammar describing all possible ways that a phrase can be expanded.

The parameters of the Markov phrase model are derived from the training corpus. The probabilities $\Pr\left(w^{(i)}|c_i\right)$ associated with the transformation of phrases into word subsequences are assigned *a priori*. In this way, a small training corpus can be used to estimated the smaller number of parameters of the Markov model without sacrificing the robustness of the overall model.

### 2.2 Corpus

In the resource management application domain, the initial training corpus consists of approximately 1200 sentences on a vocabulary of about 1000 words which are thought to be representative of the domain. These sentences were generated attempting to simulate the interaction of a person with the interactive database system. This database is further described in [5] in these proceedings.

From these initial sentences, a set of approximately 1000 sentence *patterns* were generated. This process was carried out manually. The goal was to incorporate linguistic knowledge by replacing syntactically and semantically similar components of the sentences with phrase identifiers. For

example, a typical sentence and its corresponding pattern is

> What gas surface ships which are in Coral Sea are SLQ-32 capable
> $\Longrightarrow$ *what* |*prop-type*| *surface* |*vessels*| |*optthat-are*| *in* |*water-place*| *are* |*capability*| *capable*

A phrase such as |*optthat-are*| can be expanded into the finite state grammar

$$\begin{aligned} |optthat\text{-}are| &\longrightarrow \text{(empty string)} \\ &\longrightarrow \text{which are} \\ &\longrightarrow \text{that are} \end{aligned}$$

For each experiment, these patterns were partitioned into a training and testing set. The testing set was not used in the estimation of the model parameters. The test sentences were generated from the test patterns by expanding the phrases into word sequences.

### 2.3 Parameter Estimation

For each speaker, a set of 900 training patterns was chosen which was disjoint of the patterns of the test sentences. A first order Markov model was constructed based on the training patterns (the patterns included the context of the sentence initial and sentence final boundary markers). The transition probabilities were obtained from the relative frequencies of phrases pairs in the training patterns, using a simple interpolation rule to incorporate part of the zeroth order distribution. Interpolation is used to overcome limitations of insufficient training by assigning reasonable nonzero probabilities to all event. Specifically, if $F(c_i|c_{i-1})$ is the relative frequency of $c_i$ following $c_{i-1}$ and $F(c_i)$ is the relative frequency of $c_i$ then probability of a phrase $c_i$ is assumed to be

$$\Pr(c_i|c_1,\ldots,c_{i-1}) = \lambda F(c_i|c_{i-1}) + (1-\lambda)F(c_i)$$

where in these experiments $\lambda = 0.9$ for all states. For each grammar associated with a phrase, a simple assumption that all possible word sequences are equally likely was made. Specifically, if there are $m$ different non-null expansions of a phrase $c$, then each of these expansion $w_1, \ldots w_k$ is assigned a probability

$$\Pr(w_1,\ldots,w_k|c) = (1-\theta_c)\frac{1}{m}$$

where $\theta_c$ is the probability of a null expansion. For non-optional phrases, $\theta_c = 0$.

### 2.4 Decoding Method

The decoding algorithm used to generate the results is based on the algorithm presented in [2,3]. A hidden Markov model approach is taken in which context-dependent triphone models are trained using the "forward-backward" algorithm. Whole word models are constructed by concatenation of interpolated (by context) triphone models.

The statistical language model described above is combined with these word models. Conceptually, each transition in the Markov phrase model is replaced by a network representation of the sub-grammar associated with the phrase (with branching probabilities at each of the nodes). Each arc in the grammar is replaced by the hidden Markov model for the word associated with the arc. Therefore, the entire model can be thought of a one large hidden Markov model.

The decoding algorithm attempts to find the maximum likelihood phrase sequences $c_1, \ldots c_N$ and the word expansions $w^{(i)}$ of each phrase. The output word sequence is then the concatenation of the $w^{(i)}$.

## 3 Results

Initial experiments were conducted on a speaker not included in the DARPA database in order to determine suitable system parameters (which were then unchanged).

### 3.1 Test on Training

Before evaluation on the independent test sets, two speakers were run using sentences derived from patterns in their training sets. As expected, the perplexity $Q$ [2] for the statistical model is very low in this case and recognition word error rate[3] is small. As shown in Table 1 this demonstrates

| | test on training | | | |
|---|---|---|---|---|
| speaker | MP | $(Q)$ | WP | $(Q)$ |
| dtb | 5.4% | 42.5 | 5.1% | (69.8) |
| pgh | 4.5% | 40.3 | 5.9% | (53.3) |

Table 1: Word error rate on training set (MP=Markov phrase model; WP=word pair grammar)

that when evaluated on the training set such a statistical model give low perplexity and good recognition performance. For comparison, results using a grammar (WP) is shown. This grammar is constructed to allow all two-word sequences which occur in any expansion of the training patterns. Note that even though the statistical model used incorporates the interpolation rule described above, and therefore allows all possible word sequences and not simply those in the the WP grammar, the perplexity is lower

[2] Perplexity $Q = 2^I$ where $I$ is the average information ($-\log_2 p$) of the state transitions (with probabilities $p$) in a set of sentences using a particular probabilistic model.

[3] Word error rate is the average number of substitution ($S$), deletion ($D$) and insertion ($I$) errors per reference word ($= (S + D + E)/N$ where $N$ is the number of reference words).

than the WP grammar and the performance is somewhat better.

### 3.2 Test Results

The full evaluation consisted of six speakers from the DARPA database with 25 utterance each. The word error rates are presented in Table 2. In order to evaluate

| | independent test | | test on training |
|---|---|---|---|
| speaker | MP ($Q \approx 75$) | NG ($Q = 1000$) | WP ($Q \approx 60$) |
| bef | 12.3% | 40.9% | 8.9% |
| cmr | 13.8% | 39.6% | 9.3% |
| dtb | 11.8% | 39.4% | 5.4% |
| dtd | 10.0% | 26.7% | 6.7% |
| pgh | 7.0% | 32.0% | 6.0% |
| tab | 6.3% | 21.8% | 3.2% |
| ave. | 10.2% | 33.9% | 6.6% |

Table 2: Recognition word error rate (MP=Markov phrase model; NG=null grammar; WP=word pair grammar)

the improvement due to the statistical language model, the word error rate for a "null" grammar (NG) in which all word sequences can occur is also shown. The NG result is a measure of the acoustic difficulty of the task. The result using the word-pair (WP) grammar, trained on the training *and testing* patterns is also presented in order to show that the statistical approach achieved almost equal performance without the loss imposed by imperfect coverage. Also, note that the perplexity of the statistical model ($Q \approx 75$) is comparable to the WP grammar ($Q \approx 60$)[4] despite the fact that the WP perplexity is measured on a subset of its training sentences. Finally, in order to evaluate the effect of coverage of a grammar on overall performance, consider the sentence error rates of 19% for the statistical MP case and 36% for the WP grammar. In order for the WP grammar to achieve 19% sentence error rate including the effect imperfect coverage, at least 80% of the sentences would have to parse[5]. Currently, this level of coverage is not available.

The results presented are preliminary. Several aspects of this approach have not been investigated. For instance, the structure of the Markov model has not been fully explored. Though some experiments have been performed to evaluate the use of certain higher order states which have been observed in the training, it is not clear how the model should be constructed to actually improve recognition performance significantly. Also, the assumption that all word sequences within a grammar are equally likely is clearly a very crude approximation and some improvement may be obtainable through more careful assignment of these probabilities.

[4]Perplexity on the WP grammar is obtained assuming all branches in a deterministic finite state network representation are equally likely.

[5]Suppose a fraction of sentences $x$ parse under the WP grammar. Assuming the remainder have a sentence error rate of 36%, then the overall error rate would be $(1 - x) + 0.36x$. For this to be less than 19%, $x > 80\%$

## 4 Conclusions

The results presented here demonstrate the viability of incorporating linguistic structure into a statistical model. In the resource management domain, neither solely statistical nor linguistic techniques alone are adequate at this time. Straightforward statistical techniques lack sufficient training and linguistic techniques have an inadequate coverage. However, the combination of the modest training available and simple linguistic abstractions of this training corpus provides good performance.

## References


[1] L. R. Bahl, F. Jelinek, R. L. Mercer. "A Maximum Likelihood Approach to Continuous Speech Recognition". *IEEE Trans. Pattern Analysis and Machine Intelligence*, 5(2):179-190, March 1983.

[2] Y.-L. Chow, M. O. Dunham, O. A. Kimball, M. A. Krasner, G. F. Kubala, J. Makhoul, P. J. Price, S. Roucos, R. M. Schwartz. "BYBLOS: The BBN Continuous Speech Recognition System". *IEEE ICASSP*, Dallas Texas, April 1987.

[3] F. Kubala, Y. Chow, A. Derr, M. Feng, O. Kimball, J. Makhoul, P. Price, J. Rohlicek, S. Roucos, R. Schwartz, J. Vandegrift. "Continuous Speech Recognition Results on the BYBLOS System on the DARPA 1000-Word Resource Management Database". *IEEE ICASSP*, New York, New York, March 1988.

[4] B. Lowerre, R. Reddy. "The HARPY Speech Understanding System". 1980. in *Trends in Speech Recognition*, W. A. Lea (ed.), pp. 340-360, Englewood Cliffs, N.J.: Prentice Hall.

[5] P. Price, W. Fisher, J. Bernstein, D. Pallet. "The DARPA 1000-Word Resource Management Database for Continuous Speech Recognition". *IEEE ICASSP*, New York, New York, March 1988.

[6] J. J. Wolf, W. A. Woods. "The WHIM Speech Understanding System". 1980. in *Trends in Speech Recognition*, W. A. Lea (ed.), pp. 316-339, Englewood Cliffs, N.J.: Prentice Hall.

Presented at the IEEE International Conference and Acoustics, Speech and Signal Processing, April 11-14, 1988, New York, N.Y. pp. 291-294.

# Continuous Speech Recognition Results of the BYBLOS System on the DARPA 1000-Word Resource Management Database

F. Kubala, Y. Chow, A. Derr, M. Feng*, O. Kimball, J. Makhoul,
P. Price, J. Rohlicek, S. Roucos, R. Schwartz, and J. Vandegrift

BBN Laboratories Incorporated, Cambridge, Ma. 02238

*Northeastern University, Boston, Ma. 02115

## Abstract

We present results of the BBN BYBLOS continuous speech recognition system tested on the DARPA 1000-word resource management database. The system was trained in a speaker dependent mode on 28 minutes of speech from each of 8 speakers, and was tested on independent test material for each speaker. The system was tested with three artificial grammars spanning a broad perplexity range. The average performance of the system measured in percent word error was: 1.4% for a pattern grammar of perplexity 9, 7.5% for a word-pair grammar of perplexity 62, and 32.4% for a null grammar of perplexity 1000.

## 1 Introduction

A meaningful comparison between the performance of speech recognition algorithms and systems can be made only if the systems have been tested on a common database. Even with common testing material, comparative results become difficult to interpret when grammars are used to constrain the recognition search. The ambiguity introduced by the use of grammars can be overcome by reporting results with the grammar disabled, which would establish a baseline acoustic recognition performance for the system, and by using standard generally available grammars. Finally, reporting a standard measure of the constraint provided by a grammar makes the results more meaningful.

In this paper we report results for the BBN BYBLOS system tested on a standard database using two well defined, artificial grammars and with an unconstrained null grammar. The database has been developed by the DARPA Strategic Computing Speech Recognition Program for the purpose of comparative system performance evaluation of continuous speech recognition systems [6].

In section 2, we describe the BYBLOS system. In section 3, the database and testing protocol are discussed. The grammars used in the experiments are described in section 4. Section 5 presents the recognition system results. The results are discussed in section 6.

## 2 The BYBLOS System

The BYBLOS continuous speech recognition system [2] uses discrete density hidden Markov models (HMM) of phonemes, a phonetic dictionary, and a finite state grammar to achieve high recognition performance for language models of intermediate complexity. The parameters of the HMMs are estimated automatically from a set of supervised training data. The trained phoneme models are combined into models for each word in the dictionary. These phonetic word models are then used to compute the most likely sequence of words in an unknown utterance. A formal description of a complete HMM system is presented in [1].

The BYBLOS system has been designed to accomodate large vocabulary applications. It trains a set of phoneme models which requires only a moderate amount of speech to adequately observe all the phonemes. In addition, the system trains a separate model for each distinct context in which a phoneme is observed. A phoneme's context can be defined by its adjacent phonemes or the word in which it appears. Context modeling captures coarticulation phenomena explicitly and preserves phonetic detail for those contexts which occur frequently in the training material [7]. By combining the smoothed phoneme models with the detailed context models, BYBLOS makes maximal use of the available training material. The performance improvement gained by using context dependent phoneme modeling has been reported in [3].

After training is completed, the dictionary is populated by compiling the trained phonetic models into word networks. A finite state grammar, if used, is compiled from a formal language model specification. To decode

an unknown utterance. BYBLOS utilizes the precompiled knowledge sources jointly in a time-synchronous, top-down search. This search strategy allows efficient pruning and minimizes local decisions.

BYBLOS has been demonstrated in a speaker dependent and a speaker adaptive mode. Speaker dependent modeling achieves high performance by estimating the model parameters from a training script which is large enough to contain most of the contexts likely to appear in subsequent use of the system. The speaker dependent mode has been used to achieve the results reported in this paper. The speaker adaptive mode modifies the well trained, speaker dependent word models of one speaker to model a new speaker. This technique allows the system to benefit from the well trained word models of a prototype speaker even when the training material from the new speaker is extremely limited. The adaptation mode of the BYBLOS system is discussed in [4,8].

## 3 Database

The database, described in detail in [6], was designed to provide a standard for research in speaker dependent, speaker adaptive, and speaker independent continuous speech recognition. The database was designed to cover the vocabulary, syntax, and functionality of a naval resource management task. The vocabulary consists of 1000 words. The task domain covered by the database is specified by a set of 950 sentence patterns which were used to generate the 2800 distinct sentences in the database.

The speaker dependent database provides 600 sentences (about thirty minutes of speech) designated as training material from each of twelve dialectally diverse speakers, collected in six different sessions. The scripts for the training material are designed to maximize coverage of the vocabulary and sentence patterns. The speakers include seven male and five female speakers. Independent test material was collected for the twelve speakers during additional sessions.

The experiments reported in this paper have been conducted for the purpose of comparative performance evaluation within the DARPA community. The evaluation was administered by the National Bureau of Standards (NBS). For the speaker dependent portion of the evaluation, tests were conducted using eight of the twelve available speakers.

We withheld 30 sentences from the training material for each speaker to be used for adjusting global system parameters. The remaining 570 sentences that we used for training include 952 unique words from the vocabulary. Approximately 5% of the words in the dictionary are not observed at all in the training set, 36% occur only once, and 49% occur more than once.

Twenty five sentences were selected by NBS as test material for each speaker. The test sets are different for each speaker, but on average, each set contains about 200 words. The test sentences for the eight speakers cover 46% of the dictionary, 91% of the word tokens occurring in the eight test sets have occurred more than once in the training set illustrating the effectiveness of the training data coverage over the task domain.

## 4 Grammars

The results reported below have been run using three different grammar conditions. These grammars are not intended as serious models of the task domain, but are used because they are simply defined and allow the system to be tested over a broad range of language model constraint.

A straight-forward measure of the constraint provided by a grammar is *test set perplexity* [5] which is measured on a finite state network generated by the grammar and a given set of test sentences. For the purpose of perplexity measurement, a distinguished symbol designating inter-sentence silence is added to the dictionary and to the end of each sentence of the test set. The augmented sentences are then concatenated and appended to an initial inter-sentence silence to form the word sequence, $w_1, w_2, \ldots, w_n$. If the word sequence is sufficiently long, the probability of the sequence given the grammar, $\hat{P}(w_1, w_2, \ldots, w_n)$, can be used to compute an estimate of the grammar perplexity.

The perplexity of the grammar, given the test set word sequence, is defined as:

$$L = 2^K \tag{1}$$

where

$$K = -\left(\frac{1}{n}\right)\sum_{i=2}^{n} \log_2 \hat{P}(w_i \mid w_{i-1}, \ldots, w_1) \tag{2}$$

is the average per word entropy of the language model, and

$$P(w_i) = 1 \tag{3}$$

For the grammars used in these experiments, the probabilities on the words allowed by the grammar at position $i$ in the test set word sequence are assumed to be uniform.

The three grammars, which we call the sentence pattern, word-pair, and null grammar, allow all sentences in the training and test databases. The sentence pattern grammar is compiled directly from the set of 950 sentence patterns covering all sentence types in the task domain [6]. The perplexity of the pattern grammar, averaged over the eight speakers' test sets, is 9. The word-pair grammar allows all two-word sequences allowed in the sentence pattern gram-

mar. Its perplexity is about 62. The null grammar allows all sequences of words in the vocabulary and therefore offers no language model constraint. The effective perplexity of the null grammar is equal to 1000 — the vocabulary size.

## 5 Results

The system parameters for these experiments were derived from two speakers' data collected at BBN and limited testing on two speakers from the DARPA database (CMR and BEF) using the data that we withheld from the training set. The system configuration was then fixed for the entire set of experiments. Each speaker was trained only once.

The database speech was collected at Texas Instruments (TI) in a sound isolating booth. For these experiments we used speech sampled at 20 kHz, through a Sennheiser HMD-414, close-talking, noise-canceling microphone. 14 Mel-scale-warped cepstral coefficients were computed every 10 ms, using a 20 ms data window, and vector quantized using an 8-bit codebook.

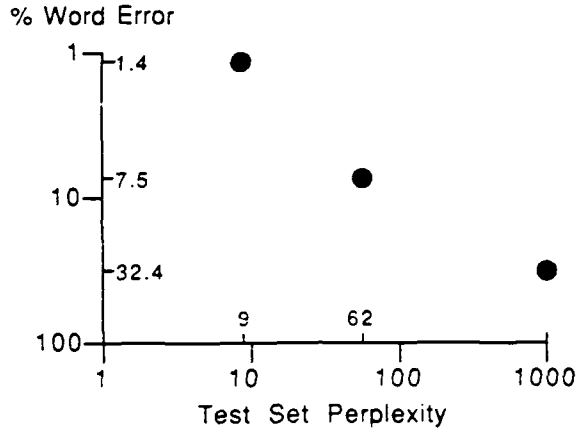



Figure 1: Recognition Performance as a Function of Grammar Perplexity. The axes are log scale.

Figure 1 shows recognition performance, averaged across the eight speakers, for the three grammar conditions. The performance is given in percent word error:

$$\text{WORD ERROR} = 100 \times (S + D + I) / N$$

where:

$S$ = number of substitution errors,

$D$ = number of deletion errors,

$I$ = number of insertion errors,

$N$ = total number of word tokens in the test sentences.

This measure has been proposed as a standard within the DARPA community. Note that since the number of insertion errors possible is not bounded, this error measure can exceed 100%.

A word hypothesis is counted in error if it does not identically match the correct word transcription. Specifically, homophones (e.g., to, two, too; or ships, ship's, ships') are counted as errors. Homophone errors typically occur only in the null grammar experiments where they account for approximately 4% of the word error rate. Furthermore, no special significance is given to errors which are phonetically close to the correct answer (minimal pair differences) or to errors which leave the semantic interpretation of the sentence intact (most deletions of the word 'the').

Individual results for each speaker are shown in Table 1. Two speakers, CMR and DTD, are female. The results are given as word error, defined above, and as word correct:

$$\text{WORD CORRECT} = 100 \times (1 - (S + D) / N)$$

where, $S$, $D$, and $N$ are defined as before.

Note that:

$$\text{WORD ERROR} \neq 100 - \text{WORD CORRECT}.$$

For the pattern and word-pair grammars, the sentence error rate and test set perplexity are also given. For the null grammar case, the sentence error rate is near 90%, and the perplexity = 1000.

## 6 Discussion

In our experience, average word error ($E$) for a set of speakers can be estimated as a function of perplexity ($L$) by:

$$E = \alpha \sqrt{L} \qquad (4)$$

Figure 1 indicates that $\alpha \approx 1$ for this data set over most of the perplexity range. We have conducted numerous experiments on speech collected at BBN in normal office environments. The experiments have used a variety of grammars including those reported here. We consistently find the average word error to be reasonably predicted by using $\alpha = \frac{1}{2}$ which is half the error rate obtained for the TI speakers. The difference in average performance between the TI and BBN data may be explained by differences in speaking style and rate. The speakers collected at BBN have some experience with speech recognition systems and generally speak more clearly than the speakers collected at TI.

While the average performance is generally predicted by perplexity, an individual speaker's performance may not be. For example, speaker DTB performs far below average for the null grammar but above average for the word-pair and pattern grammars. Similarly, the performance for RKM on the word-pair grammar is far worse than would be predicted from his results on the pattern or null grammar.

It is clear from these results that performance can be

| | Sentence Pattern | | | | Word-Pair | | | | No Grammar | |
|---|---|---|---|---|---|---|---|---|---|---|
| | word error % | word correct % | sentence error % | test set perplexity | word error % | word correct % | sentence error % | test set perplexity | word error % | word correct % |
| BEF | 2.6 | 98.3 | 20 | 8 | 8.9 | 93.2 | 44 | 62 | 40.9 | 62.6 |
| CMR | 2.7 | 99.1 | 20 | 7 | 9.3 | 94.7 | 52 | 66 | 39.6 | 65.4 |
| DTB | 0.5 | 100.0 | 4 | 10 | 5.4 | 96.5 | 32 | 34 | 39.4 | 63.1 |
| DTD | 1.0 | 99.0 | 8 | 8 | 6.7 | 94.2 | 44 | 54 | 26.7 | 75.3 |
| JWS | 0.9 | 99.1 | 8 | 9 | 4.3 | 96.2 | 28 | 59 | 25.6 | 75.4 |
| PGH | 0.5 | 99.5 | 4 | 9 | 6.0 | 96.0 | 24 | 56 | 32.0 | 70.5 |
| RKM | 2.4 | 98.1 | 16 | 10 | 16.4 | 89.7 | 52 | 64 | 30.5 | 71.8 |
| TAB | 0.5 | 100.0 | 4 | 9 | 3.2 | 97.7 | 20 | 67 | 24.8 | 76.5 |
| avg | 1.4 | 99.1 | 10.5 | 9 | 7.5 | 94.8 | 37.0 | 62 | 32.4 | 70.1 |

Table 1: Recognition Performance by Speaker for three grammar conditions.

made arbitrarily high by lowering the grammar perplexity. For large vocabulary, complex task domain applications, however, low perplexity grammars are likely to be too restrictive for real use. We expect that habitable grammars for 1000 word task domain applications will require perplexities larger than 50.

## Acknowledgement

This work was supported by the Defense Advanced Research Projects Agency and was monitored by the Space and Naval Warfare Systems Command under Contract No. N00039-85-C-0423.

## References

1 Bahl, L.R., F. Jelinek, and R.L. Mercer (1983) "A Maximum Likelihood Approach to Continuous Speech Recognition," *IEEE Trans. Pattern Analysis and Machine Intelligence* Vol. PAMI-5, No. 2, March 1983, pp. 179-190.

2 Chow, Y., M. Dunham, O. Kimball, M. Krasner, F. Kubala, J. Makhoul, P. Price, S. Roucos, and R. Schwartz (1987) "BYBLOS: The BBN Continuous Speech Recognition System," *IEEE ICASSP-87*, paper 3.7.1.

3 Chow, Yen-Lu, Richard Schwartz, Salim Roucos, Owen Kimball, Patti Price, Francis Kubala, Mari O. Dunham, Michael Krasner, John Makhoul (1986) "The Role of Word-Dependent Coarticulatory Effects in a Phoneme-Based Speech Recognition System," *IEEE ICASSP-86*, paper 30.9.1.

4 Feng, M., F. Kubala, R. Schwartz (1988) "Improved Speaker Adaptation Using Text Dependent Spectral Mappings," *IEEE ICASSP-88*, Elsewhere in these proceedings.

5 Jelinek, F. (1987) "Self-Organized Language Modeling for Speech Recognition," Unpublished manuscript, IBM T. J. Watson Research Center, Yorktown Heights, NY.

6 Price, P., W. Fisher, J. Bernstein, and D. Pallett (1988) "The DARPA 1000-Word Resource Management Database for Continuous Speech Recognition," *IEEE ICASSP-88*, Elsewhere in these proceedings.

7 Schwartz, R. M., Y. L. Chow, O. A. Kimball, S. Roucos, M. Krasner, and J. Makhoul (1985) "Context Dependent Modeling for Acoustic-Phonetic Recognition of Continuous Speech," *IEEE ICASSP-85*, paper 31.3.

8 Schwartz, Richard, Yen-Lu Chow, Francis Kubala (1987) "Rapid Speaker Adaptation using a Probabilistic Spectral Mapping," *IEEE ICASSP-87*, paper 15.3.1.

# Measuring Perplexity of Language Models Used in Speech Recognizers

Salim Roucos
BBN Laboratories
Cambridge, MA 02238

In this note, we define one measure of perplexity of a language model and present a method for computing it. We hope that by agreeing on a common method of measuring perplexity, it will become easier to compare speech recognition results when different language models are used.

## 1. Test-set perplexity

We describe a measure for characterizing the complexity of a language model; we call the measure *test-set perplexity*. This measure of perplexity is defined for any specific set of sentences and a given language model. In general, the word accuracy of a speech recognizer using a given language model is expected to decrease as the test-set perplexity of a set of test sentences increases. Knowing both the recognition performance and test-set perplexity will help in comparing recognition algorithms that use different language models.

A language model is defined by the set of probabilities $Q(w_1...w_n)$ for all word sequences $w_1...w_n$. Given a language model Q(.), the test-set perplexity of a set of sentences is defined as

$$L = 2^K \qquad (1)$$

where K, the average per word log probability (called logprob), is given by

$$K = -\,1/n\ \log_2[\,Q(w_1 w_2...w_n)] \qquad (2)$$

where $w_1...w_n$ represents the sequence of words in all the sentences of the test set, $Q(w_1...w_n)$ is the language model probability of the word sequence. The word sequence $w_1...w_n$ is obtained from a test set by concatenating all test sentences separated by sentence boundary markers. We note that the real probability of the word sequence $w_1...w_n$ is denoted by $P(w_1...w_n)$ and that we will discuss later in this note the relationship of P and Q. For the special case of a language model which assumes all the words from a vocabulary of size V are equally likely to occur at any point, i.e., $Q(w_1...w_n) = V^{-n}$, the average logprob is log V and the test-set perplexity equals the vocabulary size V for any test set from any source P(.); hence, the interpretation that test-set perplexity corresponds to the "average branching" of the language model along the test set.

## 2. Computing test-set perplexity

Equation 2 can be rewritten as

$$K = -1/n \sum_{i=1}^{n} \log_2[Q(w_i \mid w_1^{i-1})]$$

where $w_1^i$ denotes the word sequence $w_1...w_i$. In this case, we have factored the joint probability as the product of the conditional probabilities $Q(w_i \mid w_1^{i-1})$ which represent the probability that word $w_i$ will appear next given all the text up to word $w_{i-1}$. We include a special symbol to delimit sentence boundaries and this symbol counts as one word. All other delimiters (such as commas, etc.) are dropped. So, if a sentence consists of m english words, it accounts for m+1 symbols in the above logprob computation. The vocabulary includes the sentence boundary marker as one item.

The factored form is convenient for the usual finite-state N-gram models and for the deterministic grammars that have been used in speech recognizers. For the case of deterministic finite state grammars, where a word sequence is either accepted or rejected as a legal sentence of the language, it is important to make an assumption about the value of the conditional probability $Q(w_i \mid w_1^{i-1})$. In the absence of other information, we assume that all the legal words that can follow a partial word string are equally likely. To determine the number of distinct legal words that can follow a partial string of words, we need a network representation in the form of a deterministic finite state machine which means that all arcs from a node represent a possible word to follow (no null arcs) and that no two arcs leaving a node have the same word associated with them. There is a standard algorithm for converting a non-deterministic representation into a deterministic representation (see Aho & Ullman).

For more general formal languages such as context-free grammars, augmented transition network grammars, etc., one needs to determine all partial parses up to word $w_{i-1}$ and count how many distinct words can follow after word $w_{i-1}$. Then, using an assumption that all choices are equally likely, the test-set perplexity is the geometric mean of the number of word choices possible along the test set.

The factoring of the joint probability can be done in at least two directions: forward as $Q(w_i \mid w_1^{i-1})$, or backward as $Q(w_i \mid w_{i+1}^{n})$. With the deterministic finite state model and the uniform assumption of equally likely words out of a node (forward and backward), the forward and backward perplexities for the same test set of legal sentences are not the same because we have two different statistical models. Typically, we compute the perplexity of the forward (left-to-right) language model.

The same test set should be used to compute the perplexity and to measure the recognition performance. Note that perplexity depends not only on the language model but also on the particular test set (in the limit of large test sets, the variance of the test-set perplexity approaches zero).

### 3. Relation to Entropy

For a given language model, the test-set perplexity is a random variable that depends on the actual test set. For a test set $w_1 \ldots w_n$ obtained from a well behaved source (ergodic) with probability $P(w_1 \ldots w_n)$, the time average of the logprob converges to its expected value with large n:

$$\lim_{n \to \infty} -1/n \sum_{w_1 \ldots w_n} P(w_1 \ldots w_n) \log_2[Q(w_1 \ldots w_n)] = \lim_{n \to \infty} -1/n \log_2 Q(w_1 \ldots w_n)$$

where the summation is over all sequences $w_1 \ldots w_n$. Since P(.) is unknown, the right hand side is particularly useful because a large test set is sufficient to compute an estimate of the expected value. Note if Q = P, which is true when we know the correct language model, then for large n the test-set perplexity approaches the source language perplexity given by $2^H$ where H is the entropy of the language. When $Q \neq P$, the expected test-set perplexity will be larger than the language perplexity $2^H$ for any n. The goal in building language models is to minimize the difference between the expected test-set perplexity and the language perplexity. Note that for small n, K may sometimes be less than H.

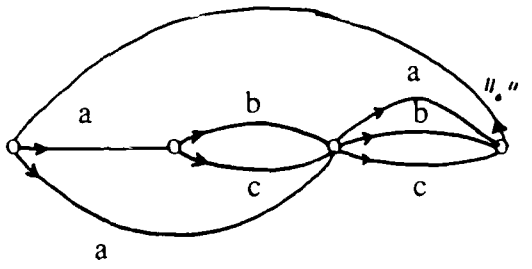
### References

A.V. Aho and J.D. Ullman, *Principles of Compiler Design*, Addison-Wesley, 1977.

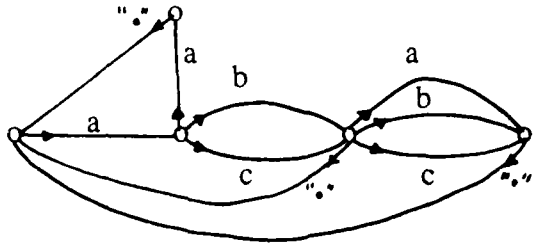
## Appendix

Given a language that allows the following 9 sentences:
aa.
ab.
ac.
aba.
abb.
abc.
aca.
acb.
acc.

One can use the following non-deterministic finite state automaton to efficiently represent the language:




To compute the test-set perplexity for the two sentences aa. and abc. we use the equivalent deterministic finite state automaton for the language:




The test-set perplexity for the two sentences "aa." and "abc." is given by:

1- concatenate as aa.abc. a seven word long test set.
2- perplexity is L= 1/7 $\log_2$(1x3x1x1x3x4x1)= 1.668

Therefore, on average the branching is 1.67 words.

Presented at the IEEE International Conference on Acoustics, Speech, and Signal Processing, May 23-26, 1989, Glasgow, Scotland.

# Iterative Normalization for Speaker-Adaptive Training in Continuous Speech Recognition

*Ming-Whei Feng*

Northeastern University, Boston MA 02115

*Richard Schwartz, Francis Kubala, John Makhoul*

BBN Laboratories, Cambridge, MA 02238

## Abstract

In this paper we present several techniques to improve the algorithm presented last year for speaker-adaptive training in continuous speech recognition. The previous method uses a transformation matrix to modify the hidden Markov Model (HMM) parameters of a pre-chosen prototype speaker to model a target speaker. To estimate the transformation matrix, it aligns a set of target speech with the same set of speech uttered by the prototype speaker using dynamic time warping. We focus on improving the previous method in the modeling of the spectral differences between two speakers, and the accuracy of the alignment. To improve the modeling of the spectral differences, we implemented a phoneme-dependent mapping procedure which transforms the prototype HMMs to the estimated target HMMs using a set of phoneme-dependent matrices. To improve the alignment, we developed a modeling of the silence, a linear duration normalization, and an iterative normalization procedure. We tested the new methods in the standard DARPA database with a grammar of perplexity 60. The performance shows a 30% word error reduction compared with that of the previous algorithm.

## 1. Introduction

Hidden Markov Modeling techniques have enjoyed great success in large-vocabulary continuous speech recognition using speaker-dependent or speaker-independent training. To achieve state-of-the-art performance in large vocabulary tasks, it has been necessary to collect a large amount of speech (~30 min) from each target speaker for speaker-dependent training, or from a large number of speakers (>100) for speaker-independent training. It is not feasible to go through such a long and tedious recording process in some applications. The speaker-adaptive training paradigm we have been advocating is designed to alleviate this difficulty. Our current method requires collecting 30 minutes of speech from only one prototype speaker, and a small amount of speech (typically two minutes) from each target speaker. Our long-term goal for speaker-adaptive training is to achieve recognition accuracy with two minutes of adaptation speech equivalent to that of 30-minute speaker-dependent training.

At ICASSP'87 [1], we presented our basic procedure for speaker-adaptive training, which uses a speaker transformation on the phonetic hidden Markov models of a prototype speaker. Starting with the trained HMM parameters of a prototype speaker and two minutes of speech from a new target speaker, we estimate a maximum likelihood probability transformation matrix by aligning the target speech against the prototype models using the forward-backward algorithm. The probability transformation matrix defining a probabilistic mapping between the prototype speaker and the target speaker is used to transform the prototype HMM models to the estimated HMM models for the target speaker.

At ICASSP'88 [2], we proposed an improved algorithm which aligns the target speech against the same sentences uttered by the prototype speaker using a dynamic time warping (DTW) algorithm to compute the co-occurring spectral pairs. We then estimate the transformation matrix by counting the spectral co-occurrences for the two speakers. This algorithm worked much better than the previous algorithm, but the performance was significantly worse for some speakers than for others. There are two possible reasons for the degraded recognition performance: (1) a single transformation matrix is not enough to model the spectral differences between the target speaker and the prototype speaker, and (2) the DTW algorithm had not found phonetically "correct" alignments between the target speech and the prototype speech, thus leading to an inferior estimate of the transformation matrices.

To improve the modeling of the spectral differences between two speakers, we implemented a phoneme-dependent transformation procedure, which uses a set of transformation matrices to transform the prototype HMM parameters to model the target speaker. Each transformation matrix represents a different probabilistic spectral mapping for each phoneme between two speakers.

We believe that the inferior alignments result from several facts. First, frequently one speaker inserts a long pause between two words in a sentence when the other speaker has not. Second, the alignment accuracy degrades when the duration of the prototype sentence is very different from that of the aligned target sentence. Third, the spectral spaces of the two speakers may be very different. To improve the alignment, we propose a modeling of the silence, a linear duration normalization before computing the alignment, and an iterative normalization procedure to compute the alignment.

### Paper Outline

In Section 2, we briefly introduce our basic speaker-adaptive training approach using phoneme-dependent mappings. In Section 3, we propose several techniques to improve the alignment: a modeling of the silence in the target speech and a linear duration normalization before computing the alignment, and an iterative normalization algorithm to compute the alignment and estimate the mappings. We show that this algorithm converges to a local minimum of the mean-squared error for the alignment. To evaluate the proposed algorithm, we then present in Section 4 speaker-adaptive recognition results using two minutes of target speech on the standard DARPA database, compared to the performance of 28-minute speaker-dependent training.

## 2. Phoneme-Dependent Mappings

In this section we describe our speaker adaptation procedure using phoneme-dependent mappings. For each state of a discrete HMM, we have a discrete probability density function (pdf) defined over a fixed set N of spectral templates. We can view the discrete pdf for each state $s$ as a probability row vector $\underline{p}(s)=[p(k_1/s),p(k_2/s),...,p(k_N/s)]$ where $p(k_i/s)$ is the probability of spectral template $k_i$ at state $s$ of the HMM model.

If we denote a quantized spectrum from the prototype speaker as $k_i$, $1 \le i \le N$, and from the target speaker as $k'_j$, $1 \le j \le N$, where $i$ and $j$ are indices denoting the quantized spectra, then we can denote the probability that the target speaker will produce quantized target spectrum $k'_j$, given the prototype speaker produced spectrum $k_i$, as $p(k'_j/k_i)$ for all $i,j$. The probabilistic mapping can be defined as follows:

$$p(k'_j/s)=\sum_{i=1}^{N} p(k_i/s)p(k'_j/k_i),\ 1 \le j \le N \qquad (1)$$

The probabilities $p(k'_j/k_i)$ for all $i$ and $j$ form an NxN matrix, T, which can be interpreted as a probabilistic transformation matrix from one speaker's spectral space to another's at each state. We can then compute the discrete pdf $\underline{p}'(s)$ at state s for the target speaker as the product of the row vector $\underline{p}(s)$ and the matrix T

$$\underline{p}'(s)=\underline{p}(s) \times T, \text{ where } T_{ij}=p(k'_j/k_i) \qquad (2)$$

In equations (1) and (2), we assume that the probability for spectrum $k'$ given $k$ is independent of $s$, which indicates that a single probabilistic spectral mapping will transform the speech of one speaker to another. However in practice, some of the differences between speakers can not be modeled by a single transformation. To have a more detailed modeling of the spectral differences between two speakers, we define a phoneme-dependent mappings:

$$p(k'_j/s)=\sum_{i=1}^{N} p(k_i/s)p(k'_j/k_i,\phi(s)) \qquad (3)$$

where $\phi(s)$ specifies an equivalence class of states in models that represent the same phoneme as $s$.

Since only two minutes of target speech is available, it is not adequate to estimate reliable probabilistic mappings for all phonemes. Therefore, in transforming the HMM models, we *interpolate the phoneme-dependent mappings* with the phoneme-independent mappings to improve the robustness of the adapted HMM models. The weight for the combination is different for each prototype spectral index (each row of the transformation matrices), and it is dependent on the number of occurrences of the observed prototype spectrum for that phoneme in the adaptation speech. The important step in the phoneme-dependent mapping procedure is to estimate the $p(k'_j/k_i,\phi(s))$ that optimizes the recognition performance. In next section, we will describe an iterative algorithm to estimate the phoneme-dependent mappings.

## 3. Improving the Alignment

We present several techniques to improve the alignment accuracy. In Section 3.1 we introduces two processes before performing the alignment. Then in Section 3.2 we describe a new iterative algorithm to compute the alignment. We deal with the convergence of the iterative algorithm in Section 3.3. .

### 3.1 Before Computing the Alignment

Insertion of Silence into Prototype Speech

When two different speakers read the same sentence, they may insert pauses (silence) in different places of the sentence. In this case, the DTW is forced to align silence frames to speech frames resulting in phonetically incorrect alignments. To achieve correct alignments for target speech with arbitrary inter-word pauses, we insert a synthesized silence spectrum between each word of the prototype speech. We compute the synthesized silence spectrum for each utterance as the average of the spectral parameters over several frames of silence from the target speaker.

Linear Warping the Target Spectra

The warping function produced by DTW can be viewed as a mapping from the time axis of one pattern onto that of another. To align two sets of spectra with the same text, it is reasonable to use DTW with slope constraints [5], so as to avoid unrealistic mappings, for example that one target frame gets aligned to many prototype frames. However if the durations of the two aligned sentences are very different, the slope-constraint may interfere with their alignment accuracy. Since our prototype speaker speaks slowly and carefully, large speaking rate differences can exist in words or word sequences which are spoken rapidly and casually by the target speakers. To overcome this problem, we linearly warp the spectral sequence of each target sentence to be the same length as the corresponding prototype spectral sequence. We perform the linear warping on the target spectral sequence by copying the target spectral frame that is closest to a linear time scaling. It avoids losing detailed information in the original target spectral sequence due to interpolation.

### 3.2 Computing the Alignment Iteratively

When the target spectral space is very different from the prototype spectral space, the minimum error alignment produced by the DTW algorithm may not correspond to the phonetically correct alignment. To improve the accuracy of the alignment, we developed an iterative alignment and normalization procedure. In each iteration of the algorithm, we align the target speech to the prototype speech; then using the alignment we shift each target spectral frame in the target speech by an amount that is dependent on the index of the corresponding vector-quantized value of the prototype frame to which it aligns. Using the shifted values of the target frames, we realign the target speech to the prototype speech, etc. We will prove in Section 3.3 that this algorithm converges to a local minimum in the mean-squared error (mse) for the alignment. The mse of a given alignment is equal to the average of the squared-difference of the target spectral frames and the corresponding prototype spectral frames in the alignment path. The formal definition of the mse given an alignment will be defined later in equation (4). The detailed algorithm is describe as follows:

1. Classify (quantize) each frame of the prototype spectra into one of $2^M$ VQ regions using a well-trained M-bit prototype VQ codebook.
2. Align each target spectrum $\{x\}$ in the adaptation sentences to a prototype spectrum $\{y\}$ using a slope-constrained DTW. If the mse of the alignment is similar to the mse of the alignment resulting from the previous iteration, then the algorithm converges and we stop.

3. Classify each frame of the target spectra into one of the prototype VQ regions according to the classification of its aligned prototype frame.
4. Compute the mean of $\{x\}$ and $\{y\}$ in each of the prototype VQ regions.
5. Shift each frame of the target spectra by the difference vector between the mean of $\{x\}$ and $\{y\}$ for that VQ region.
6. Substitute the target spectra by the shifted target spectra, and go to step 2.

Estimation of Phoneme-Dependent Mappings

From the alignment of the last iteration, we count the co-occurrences of the quantized spectral indices for each of the frames in the adaptation sentences, and form the co-occurrence matrix $N(\phi(s))$ for each phoneme, where each element $N_{ij}$ is the number of the co-occurrences of the target spectra $k'_j$ and the prototype spectra $k_i$. Then we normalize the rows of $N(\phi(s))$ to form the phoneme-dependent transformation matrices $T(\phi(s))$.

Below we prove that the iterative algorithm is guaranteed to converge to a local minimum in the mse of the alignment.

### 3.3 Convergence of the Iterative Algorithm

DTW Minimizes the Mse

Suppose $\{x_i, 1 \le i \le I\}$ is an I-frame target spectral sequence for a sentence and $\{y_j, 1 \le j \le J\}$ is a J-frame prototype spectral sequence for the same sentence. As a measurement of the difference between two feature vectors $x_i$ and $y_j$, a Euclidean distance $d(c)=d(i,j)=\|x_i-y_j\|$ is employed between them. Using a dynamic time warping (DTW) algorithm [5], we obtain a warping function $F = c(1),c(2),....,c(k),....c(K)$, where $c(k)=(i(k),j(k))$, by minimizing the accumulated mse between $\{x\}$ and $\{y\}$, which is

$$E(F)=(1/K)\sum_{k=1}^{K}[d(c(k))^2]=(1/K)\sum_{k=1}^{K}\|x_{i(k)}-y_{j(k)}\|^2 \qquad (4)$$

The warping function, which is also called the alignment, realizes a mapping from the time axis of $\{x_i, 1 \le i \le I\}$ on that of $\{y_j, 1 \le j \le J\}$. The alignment indicates the matched prototype spectral frame given each target spectral frame.

Shifting Target Spectrum Reduces mse Further

Given the alignment (the paired target spectral frames and prototype spectral frames), we can reduce the mse of the alignment further by making the target spectral space closer to the prototype spectral space. Suppose we shift each target spectral frame by a single vector $x_0$. The mse of the alignment becomes

$$E(F)=(1/K)\sum_{k=1}^{K}\|(x_{i(k)}-x_0)-y_{j(k)}\|^2$$

$$=(1/K)\sum_{k=1}^{K}\{x_{i(k)}^2+x_0^2+y_{j(k)}^2+$$

$$2x_0y_{j(k)}-2x_{i(k)}x_0-2x_{i(k)}y_{j(k)}\} \qquad (5)$$

By taking the partial derivative of E(F) with respect to $x_0$, we have

$$\partial E(F)/\partial x_0=2x_0-(2/K)\sum_{k=1}^{K}x_{i(k)}+(2/K)\sum_{k=1}^{K}y_{j(k)} \qquad (6)$$

By setting equation (6) equal to zero, we obtain

$$x_0=(1/K)\sum_{k=1}^{K}x_{i(k)}-(1/K)\sum_{k=1}^{K}y_{j(k)} \qquad (7)$$

Therefore shifting the target spectral frames by the difference of the means of $\{x\}$ and $\{y\}$ minimizes the total mse of the given alignment.

It is very possible that the mean difference between two spectral spaces varies across different VQ partitions. To make a more complex modeling of two spectral spaces, we shift the target spectrum using a set of VQ-dependent vectors $\{x(1),x(2),...,x(2^M)\}$. The total mse can be represented by the following equation

$$E(F)=\sum_{l=1}^{2^M}(1/K(l))\sum_{k=1}^{K(l)}\|(x_{i(k)}^{(l)}-x(l)-y_{j(k)}^{(l)})\|^2 \qquad (8)$$

By following the same approach as described above, we achieve

$$x(l)=(1/K(l))\sum_{k=1}^{K}x_{i(k)}^{(l)}-(1/K(l))\sum_{k=1}^{K}y_{j(k)}^{(l)},\ 1 \le l \le 2^M \qquad (9)$$

which minimize both the total mse and the mse for each VQ region.

Convergence of Mse

In the first iteration, step two of the algorithm produces the first alignment between the prototype speech and the target speech by minimizing the mse. Shifting the target spectra at step five by the mean difference reduces the mse at each VQ region independently and also gives the minimum mse for that alignment.

After the first iteration, we realign the shifted target spectra with the original prototype spectra, and the obtained alignment may or may not be different from the previous alignment. If the alignment is different, then the current mse must be smaller than that of the previous iteration because DTW provides a new alignment with the smallest mse. If the alignment is the same as the previous one, then the mse is not changed and the algorithm has converged. In the next section, we present some experimental results by using the iterative algorithm with three iterations.

## 4. Experimental results

### 4.1 Experimental Conditions

In the experiments shown below, we use the well-trained HMMs of a single speaker RS as the prototype. RS is a careful male speaker with a New York dialect. RS recorded 600 sentences at BBN in normal office environment. The 600 utterances constituted about 30 minutes of speech which was used to estimate the HMM parameters for the prototype models.

A 1000-word database of continuously read speech has been designed and recorded within the DARPA Strategic Computing Speech Recognition Program [3]. This data consists of read sentences which are appropriate in a naval resource management task. It was recorded in a sound-isolated recording booth at Texas Instruments (TI). We use eight TI speakers from this database to test different adaptation procedures compared with the performance of the 28-minute speaker-dependent training.

In the adaptation experiments of this section, we use adaptation material of two minutes duration. We lowpass filtered both the target speech and the prototype speech at 10 kHz and sampled at 20 kHz. Every sentence in the target speech and the prototype speech is represented by frames of mel-frequency cepstral coefficients (MFCCs).

All the recognition experiments used a word-pair grammar of perplexity 60. This grammar allows all two-word sequences which occur in the task domain definition [4]. The recognized sequence of words was compared automatically to the correct answer to determine the percentage of errors of each type: substitutions, deletions, and insertions. We use an error measure that reflects all three types of errors in a single number. The percent error is given by

$$\%word\ error = 100 \times \frac{(substitutions + insertions + deletions)}{total\text{-}input\text{-}words}$$

### 4.2 Results

| Experiments | % Word Error |
|---|---|
| Baseline (ICASSP 88) | 13.8 |
| Phoneme-Dependent Mappings | 12.1 |
| Iterative Normalization Algorithm | 9.6 |
| 28-Min Speaker-Dependent Training | 7.1 |

**Table I. Percent word recognition error for 2-min speaker-adaptive training vs. 28-min speaker-dependent training on 8 speakers**

Table I contains the recognition word error rates using different adaptation procedures with two minutes of target speech, in comparison with the 28-minute speaker-dependent performance on eight TI speakers. Below we discuss the performance of each algorithm, row by row, in Table I.

In the first row, we show the performance for our previous algorithm [2], which uses a single transformation matrix (phoneme-independent transformation) to transform the prototype HMM models to the target HMM models. The word error rate is 13.8% which is about two times that of the 28-minute speaker-dependent performance.

By using a set of phoneme-dependent transformation matrices (instead of one matrix) to perform the transformation, we reduce the word error rate to 12.1% (shown in the second row of Table I). Compared to the performance of the previous algorithm, it resulted in a 12% error reduction.

By applying the silence modeling, the linear duration normalization, and the iterative normalization algorithm described in Section 3 to improve the alignment and estimate the phoneme-dependent mappings, we obtain a significant improvement from word error 12.1% to 9.6%, which is a 20 percent reduction in the word error rate.

The speaker-dependent performance with 28 minutes of training speech shown in table I is 7.1% word error using steady-state MFCCs only. The performance has been improved to 3.6% word error recently by using additional feature parameters, including differential MFCCs and power parameters. We believe that further improvement on speaker-adaptive training can be achieved by using more features.

## 5. Conclusions

In this paper we presented several techniques to improve the algorithm presented last year for speaker-adaptive training. The previous method uses a transformation matrix to modify the hidden Markov Model (HMM) parameters of a pre-chosen prototype speaker to model a target speaker. It estimates the transformation matrix by aligning a set of target speech with the same set of speech uttered by the prototype speaker using DTW. We focus on improving the previous algorithm by: (1) modeling the spectral differences between two speakers using a set of phoneme-dependent transformation matrices, and (2) improving the alignment accuracy by using a modeling of the silence, a linear duration normalization, and an iterative alignment procedure. To evaluate the effectiveness of the new methods, we performed experiments on the standard DARPA database with a grammar of perplexity 60. The recognition performance of the new algorithm shows a 30% word error reduction compared with that of the previous algorithm.

## Acknowledgement

This work was supported by the Defense Advanced Research Projects Agency and was monitored by the Space and Naval Warfare Systems Command under contract number N00039-85-C-0423.

## References

[1].R. Schwartz, Y.L. Chow, and F. Kubala, "Rapid Speaker Adaptation Using a Probabilistic Spectral Mapping", IEEE Int. Conf. Acoust., Speech, Signal Processing, Dallas,TX, April 1987, pp. 633-636.

[2].M. W. Feng, F. Kubala, R. Schwartz, and J. Makhoul, "Improved Speaker Adaptation Using Text-Dependent Spectral Mappings", IEEE Int. Conf. Acoustic Speech Signal Processing, New York, April 1988, pp. 131-134.

[3].P. Price, W. Fisher, J. Bernstein, and D. Pallett, "The DARPA 1000-Word Resource Management Database for Continuous Speech Recognition ", IEEE Int. Conf. Acoust., Speech, Signal Processing, New York, April 1988, pp 651-654.

[4].F. Kubala, Y. Chow, A. Derr, M.W. Feng, O. Kimball, J. Makhoul, P. Price, J. Rohlicek, S. Roucos, R. Schwartz, and J. Vandergrift, "Continuous Speech Recognition Results of the BYBLOS System on the DARPA 1000-Word Resource Management Database", IEEE Int. Conf. Acoust., Speech, Signal Processing, New York, April 1988, pp. 291-294.

[5].H.Sakoe and S. Chiba, "Dynamic Programming Algorithm Optimization for Spoken Word Recognition", IEEE Trans. on Acoustics, Speech, and Signal Processing, Vol. ASSP-26, No.1, Feb. 1978.